Universidade de Aveiro
2012
Departamento de Electrónica, Telecomunicações e Informática

Ana Patrícia Oliveira Ferreira da Rocha

MonitorMe: Sistema de Reconhecimento de Atividades baseado em Android

MonitorMe: Android-based Online Activity Recognition System

Universidade de Aveiro
2012

Departamento de Electrónica, Telecomunicações e Informática

Ana Patrícia Oliveira Ferreira da Rocha

MonitorMe: Sistema de Reconhecimento de Atividades baseado em Android

MonitorMe: Android-based Online Activity Recognition System

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor José Maria Amaral Fernandes, Professor Auxiliar do Departamento de Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro e do Doutor João Paulo Trigueiros da Silva Cunha, Professor Associado com Agregação do Departamento de Engenharia Eletrotécnica e de Computadores da Universidade do Porto.

o júri / the jury

presidente / president

Prof. Doutora Maria Beatriz Alves de Sousa Santos
Professora Associada com Agregação do Departamento de Electrónica Telecomunicações e Informática da Universidade de Aveiro

vogais / examiners committee

Prof. Doutora Ana Cristina Costa Aguiar
Professora Auxiliar Convidada do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Prof. Doutor José Maria Amaral Fernandes
Professor Auxiliar do Departamento de Electrónica Telecomunicações e Informática da Universidade de Aveiro

Prof. Doutor João Paulo Trigueiros da Silva Cunha
Professor Associado com Agregação do Departamento Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

agradecimentos / acknowledgments

First and foremost, I would like to extend my thanks and gratitude to my supervisors, Professor Doctor José Maria Fernandes and Professor Doctor João Paulo Cunha, for their guidance and advice throughout the entire execution of this dissertation.

My sincere thanks also go to all my colleagues at IEETA and DETI, for the helpful discussions and for their availability for the experimental tests. Without their collaboration most of the work presented in this dissertation would not be possible.

I would like also to thank Serviços de Acção Social da Universidade de Aveiro, for allowing the use of the University of Aveiro track and field facility. To Bartlomiej Steidel, for video related discussions. And to Eng. David Ribeiro and Eng. Óscar Pereira for their availability and help concerning the use of the MARG module and in the demonstration.

Finally to my family and friends, for all of their interest and support.

palavras-chave

monitorização, reconhecimento de atividades, *smartphone* Android, vídeo, cadência de imagens variável, estimativa de velocidade, sensores, acelerómetro, giroscópio.

resumo

A monitorização de uma pessoa pode ser importante em várias situações do dia-a-dia. Um modo de monitorização é a identificação de atividades realizadas. Atualmente, vários sensores potencialmente úteis para o reconhecimento de atividades, são integrados em dispositivos móveis, o que os torna particularmente interessantes para este tipo de monitorização.

Uma forma complementar de monitorização é a utilização da gravação de um vídeo do ambiente que rodeia a pessoa a ser monitorizada. No entanto, dado o tamanho elevado dos vídeos para transmissão por canais sem fios ou mesmo para gravação no dispositivo, torna-se necessário atuar na compressão e redução da informação associada. Uma forma de o conseguir é adaptar a cadência de imagens adquiridas à velocidade da pessoa que está ser monitorizada.

Nesta dissertação é proposto um sistema de monitorização *online*, chamado MonitorMe, que permite o reconhecimento de atividades e a gravação de um vídeo do ambiente envolvente de uma pessoa. Este sistema inclui um *smartphone* Android, mantido num bolso de camisa, e um módulo MARG (*Magnetic, Angular Rate and Gravity*), colocado num bolso das calças. Foi desenvolvida uma aplicação para o *smartphone*, que obtém dados dos sensores integrados em ambos os dispositivos para a realização do reconhecimento *online* de 6 atividades diferentes (em pé, sentado, deitado, andar, correr e queda). Este reconhecimento é conseguido utilizando um algoritmo de baixo custo computacional, cujo desenvolvimento teve em consideração as restrições relativas à capacidade de processamento e à duração da bateria dos telemóveis.

Paralelamente ao reconhecimento de atividades, a câmara do *smartphone* captura imagens com uma cadência que varia com a velocidade do utilizador, esta última estimada a partir dos dados dos sensores processados para o reconhecimento de atividades. Demonstra-se assim a possibilidade de, com baixo custo computacional, diminuir a largura de banda de transmissão ou o armazenamento no dispositivo móvel.

O sistema MonitorMe foi treinado e depois testado com dados obtidos em duas experiências envolvendo 10 pessoas, num total de 440 eventos diferentes com uma duração total de 45 minutos (2/3 usados para treino e 1/3 para teste). Os resultados globais obtidos mostraram uma sensibilidade superior a 93% e uma especificidade superior a 98% para o reconhecimento de atividades, e um erro médio relativo de 8.6% para a estimativa de velocidade.

keywords

online monitoring, activity recognition, Android smartphone, video, variable frame rate, speed estimation, sensors, accelerometer, gyroscope.

abstract

The monitoring of a given person can be important in different day-to-day scenarios. Monitoring can be performed by detecting activities while being carried out. Presently, various sensors with potential for activity recognition are being included in mobile devices, so they are particularly interesting for this type of monitoring.

A complementary way of monitoring consists in the use of a video recording of the subject's surrounding environment. However, given the large size of the videos for transmission through wireless links or even for storage in the device, it is necessary to compress and reduce the corresponding information. This can be achieved by adapting the frame rate of the captured images to the speed of the user being monitored.

In this dissertation an online monitoring system, MonitorMe, which performs activity recognition and video recording of the surrounding environment of a subject, is proposed. This system includes an Android smartphone, inserted in a shirt pocket, and an MARG (Magnetic, Angular Rate and Gravity) module, placed in a pants pocket. A smartphone application was developed, which collects data from the sensors integrated in both devices to perform the online recognition of 6 different activities (standing, sitting, lying, walking, running and fall). This was achieved by using an algorithm of low computational cost, which took into account the existing restrictions regarding processing power and battery life of mobile phones.

In parallel with activity recognition, the smartphone camera captures images with a frame rate that varies with the user speed, the latter estimated from sensor data processed for activity recognition. This demonstrates the possibility of reducing the required transmission bandwidth or the storage in the mobile device, with a low computational cost.

The MonitorMe system was trained and then tested using data collected in two experiments with a participation of 10 subjects, which resulted in a total of 440 different events with a total duration of 45 minutes (2/3 used for training and 1/3 for testing). The overall results have shown a sensibility greater than 93% and a specificity greater than 98% for activity recognition, and an average relative error of 8.6% for speed estimation.

Contents

List of Figures iii
List of Tables vi
List of Abbreviations and Acronyms vii
1 Introduction 1
1.1 Motivation and Context 1
1.2 Objectives 2
1.3 Contributions 3
1.4 Dissertation Structure 3
2 Background and Related Work 5
2.1 Activity Recognition 5
2.2 Speed Estimation 15
2.3 Video Monitoring 17
3 MonitorMe System 19
3.1 Monitoring System 19
3.2 MonitorMe Architecture 22
3.3 MonitorMe Implementation 25
3.3.1 Programming Environment 25
3.3.2 Android Application 26
4 MonitorMe Algorithm 33
4.1 Data and Methods 33
4.2 Activity Recognition 36
4.2.1 Algorithm 37
4.2.2 Data Analysis 38
4.3 Speed Estimation 52
4.3.1 Video Control 53
4.3.2 Data Analysis 53
4.4 Experimental Results 58
4.4.1 Activity Recognition 59
4.4.2 Speed Estimation 63
5 MonitorMe Demonstration 67
6 Conclusions and Future Work 73
6.1 Conclusions 73

6.2 Future Work 74
References 75

List of Figures

Figure 1. Sequence of steps usually needed for activity recognition. 5

Figure 2. Values for the phone accelerometer (a) and gyroscope (b) on the three axes, when changing between three static positions. 7

Figure 3. Values for the phone accelerometer (a) and gyroscope (b) on the three axes, when dropping from a static position..... 8

Figure 4. Use case diagram of an online monitoring system that performs activity recognition. 20

Figure 5. Activity diagram that includes the various tasks necessary for online activity recognition. 20

Figure 6. Activity diagram of the MonitorMe smartphone application. 21

Figure 7. The MonitorMe system architecture, including the location of the smartphone and the MARG module device (and associated coordinate systems). 22

Figure 8. Axes orientation for the: (a) smartphone accelerometer; (b) smartphone gyroscope. 23

Figure 9. Axes orientation for the: (a) MARG module accelerometer; (b) MARG module gyroscope. 25

Figure 10. Package diagram of MonitorMe Android application. 27

Figure 11. Class diagram of MonitorMe Android application. 27

Figure 12. Capture of the smartphone screen when running the MonitorMe Android application.. 28

Figure 13. Example of a sequence performed in experiment 1: (a) activity being carried out at each instant; (b) data obtained from the phone accelerometer, phone gyroscope, module accelerometer and module gyroscope, respectively. 34

Figure 14. Capture of the smartphone screen when running the Android application for the sensors data collection. 35

Figure 15. Decision tree that represents the activity recognition algorithm..... 39

Figure 16. Box plot for the minimum value considering all three axes, according to (1), for the phone accelerometer..... 41

Figure 17. Box plot for the minimum value considering all three axes, according to (1), for the module accelerometer. 41

Figure 18. Box plot for the maximum value considering all three axes, according to (2), for the phone accelerometer..... 42

Figure 19. Box plot for the minimum value considering all three axes, according to (1), for the phone gyroscope..... 42

Figure 20. Box plot for the maximum value considering all three axes, according to (2), for the phone gyroscope..... 42

Figure 21. Box plot for the sum mean square values, according to (3), for the phone accelerometer. 44

Figure 22. Box plot for the sum mean square values, according to (3), for the phone gyroscope. .. 44

Figure 23. Box plot for the sum of variances values, according to (4), for the phone accelerometer. 45

Figure 24. Box plot for the sum of variances values, according to (4), for the module accelerometer. 45

Figure 25. Box plot for the sum of mean absolute differences values, according to (5), for the phone accelerometer. 46

Figure 26. Box plot for the mean values on each axis of the phone accelerometer, for walking and running. 47

Figure 27. Mean values on the three axes of the phone accelerometer, for standing, sitting and lying postures..... 48

Figure 28. Mean values on the three axes of the module accelerometer, for standing and sitting postures..... 49

Figure 29. Example of how the transitions are dealt with in MonitorMe. 50

Figure 30. Range of values obtained for the sum of minimums on each axes of the phone accelerometer, according to (6), considering falls and transitions confused with falls. 51

Figure 31. Range of values obtained for the sum of maximums and minimums on each axis axes of the phone accelerometer, according to (8), considering falls and transitions confused with falls. .. 52

Figure 32. Scatter plot of the average speed versus the sum of mean squares on each axis of the phone accelerometer, including the corresponding linear regression. Points A and B refer to two different events. In event A a subject was running with higher speed than the speed corresponding to the subject in event B. 54

Figure 33. Scatter plot of the average speed versus the sum of mean squares on each axis of the module accelerometer, including the corresponding linear regression. 54

Figure 34. Scatter plot of the average speed versus sum of mean absolute differences on each axis of the phone accelerometer, including the corresponding linear regression. 55

Figure 35. Scatter plot of the average speed versus sum of mean absolute differences on each axis of the module accelerometer, including the corresponding linear regression. 55

Figure 36. Scatter plot of the average speed versus the sum of variances on each axis of the phone accelerometer, including the corresponding linear regression. Points A and B refer to two different events. In event A a subject was running with higher speed than the speed corresponding to the subject in event B. 56

Figure 37. Scatter plot of the average speed versus the sum of variances on each axis of the module accelerometer, including the corresponding linear regression. 56

Figure 38. Scatter plot of the average speed versus the sum of variances on each axis of the phone gyroscope, including the corresponding linear regression. 57

Figure 39. Scatter plot of the average speed versus the sum of variances on each axis of the module gyroscope, including the corresponding linear regression. 58

Figure 40. Estimated average speed versus the measured average speed for the test set of experiment 2 (walking and running). 63

Figure 41. Activities recognized at each instant during the demonstration of the MonitorMe. The actual activities are shown above the graph, using the notation indicated in the graph y-axis. The red dashed lines indicate the duration of each actual activity. The read ellipses show incorrect recognition during transitions..... 67

Figure 42. Capture from the remote device screen (a) and from the video used for identification of the actual activities (b), while the subject was sitting. 69

Figure 43. Capture from the remote device screen (a) and from the video used for identification of the actual activities (b), while the subject was walking. 69

Figure 44. MonitorMe demonstration of the adaptable video frame rate: the frame rate value is computed according to activity/speed (a); this value is used to adjust the time interval between consecutive captured frames (b). The alphanumeric symbols on the top of figure (a) represent the activity being carried out in each time interval. The red ellipse corresponds to a case analyzed in detail in the text. 70

List of Tables

Table 1. State of the art summary of activity recognition research 11

Table 2. State of the art summary of speed estimation research 16

Table 3. Samsung Galaxy S II main specifications 24

Table 4. Users' characterization in both experiments, including the average and [minimum, maximum] range values for age, weight, height and BMI 33

Table 5. Activity distribution in the training and test sets, represented by the number of events and the duration in seconds 36

Table 6. Features used in the activity recognition algorithm, including the corresponding expression and notation 37

Table 7. Features and thresholds used in the activity recognition algorithm 38

Table 8. Feature values for two subjects corresponding to points A and B of Figure 32 and Figure 36 57

Table 9. Confusion matrix when using the single continuous file with the activity events concatenated in random order * 60

Table 10. Sensitivity, specificity and accuracy for each activity, when using the single continuous file with the activity events concatenated in random order 60

Table 11. Confusion matrix when considering the activity events separately * 61

Table 12. Sensitivity, specificity and accuracy for each activity, when considering the activity events separately 61

List of Abbreviations and Acronyms

The abbreviations/acronyms used only once in this dissertation are not included in the following list. They are defined in the text.

2/3/4G	$2^{nd}/3^{rd}/4^{th}$ generation mobile telecommunications
ADL	Activities of Daily Living
API	Application Programming Interface
BMI	Body Mass Index
dps	Degrees Per Second
fps	Frames Per Second
GPS	Global Positioning System
HMM	Hidden Markov Models
IDE	Integrated Development Environment
JPEG	Joint Photographic Experts Group
MARG	Magnetic, Angular Rate and Gravity
MSE	Mean Squared Error
NN	Neural Networks
OS	Operating System
RMS	Root Mean Square
SDK	Software Development Kit
UI	User Interface
Wi-Fi	Wireless Fidelity

1 Introduction

1.1 Motivation and Context

The monitoring of a given person can be very important in some areas and professions. Monitoring can be defined as the action of supervising the state of something or someone, normally over a period of time and for a special purpose [1].

Regarding elderly citizens, the population of the age group 65 and over has been increasing worldwide, and this growth is expected to continue at least until 2100 [2]. A monitoring system can enable the supervision of elders living alone and also of patients recovering at home. In the case of specific diseases, such as Parkinson, monitoring can also be used to study the disease related movements and the improvements due to medication.

Monitoring is also potentially useful in others situations related to healthcare. In 2008, around 47% of the adults worldwide, aged 20 years old and over, were overweight or obese (35% and 12% respectively) [3]. The scenario is even worse if we consider that childhood obesity has been increasing [4]. A monitoring application can be used not only to study lifestyles, but also to help improve these figures and prevent obesity related diseases. This may be achieved by generating reports with information obtained through monitoring, which can be for example sent to the user's doctor. This kind of reports is also very useful in the area of professional sports, where exercise information can be used to present a graph with the evolution of the athlete performance across training sessions.

Some professions where monitoring can be very important, include professionals in the field (e.g. first responders). In these cases, a monitoring application can allow the identification of life threatening situations. There are also other professions, such as in the security area, where the continuous monitoring of the worker's surrounding environment and his/her type of activity can be of enormous use.

In the situations describe above as well as in other scenarios, monitoring can be achieved in many different ways. When monitoring people, one way consists in the use of a video recording of the person to be monitored. However, when the person's location changes over time, recorded images of the surrounding environment (obtained with a wearable camera, for example) are more appropriate.

One potential problem with video utilization, in the context of online monitoring, is that standard video requires considerable processing power, by the devices sending/receiving the corresponding data, and a large transmission bandwidth. To alleviate these requirements, a possible solution consists in adapting the video frame rate according to the monitoring scenario. This can imply finding the adequate frame rate value for a given situation (e.g. a low speed of the monitored subject may correspond to a lower frame rate and a high speed to higher frame rate).

In the monitoring scenario of a given person, the identification of the activity changes can also be very useful. This information and an adaptable image recording of the surrounding environment can be used together in a monitoring solution. For example, considering online monitoring, the

recorded images can help in verifying an identified activity, such as a fall, allowing an appropriate and timely response.

1.2 Objectives

In the context just described, the main objective of this dissertation is to implement an online monitoring system, named MonitorMe, which recognizes activities based on sensors and records a video of the surrounding environment. MonitorMe is based on an Android smartphone with a camera and sensors. This device is to be placed at the chest (e.g. in a shirt pocket) and should be able to exchange information with other devices through data connections, such as a Bluetooth or a Wi-Fi (Wireless Fidelity) link. In this study one of these devices is an in-house built module based on MARG (Magnetic, Angular Rate and Gravity) sensors [5] placed at the upper right leg (e.g. in the pants pocket).

The activities to be recognized by this system include the following: standing, sitting and lying (static activities or postures), walking and running (dynamic activities), and fall (instantaneous activity). The use of another device, besides the smartphone, aims at helping the recognition of one of the static activities (the sitting posture). Indeed it would be very difficult to distinguish this posture from the standing posture with just one device placed at the chest, since the position of the trunk can be similar in both postures.

Since the system is based mainly on a smartphone to be used on a daily basis, the activity recognition approach used in our system aims at achieving a balance between accuracy and required processing resources. Also, the impact of each device and each sensor on the quality of the proposed system is investigated.

Regarding the video of the surrounding environment, to minimize the use of the smartphone resources and the required transmission bandwidth, the frame rate is set according to the user speed while walking or running. To estimate this speed, instead of using the phone GPS (Global Positioning System) that represents extra resources and is only available outdoors, the possibility of using information already extracted from the sensor data for activity recognition is studied.

The smartphone should be able to send to a server the current identified activity and the images being recorded. A remote monitoring device can connect to the server to receive this data and present it to its user. So it is necessary to develop applications for an Android smartphone, a server and a remote device. The main application of MonitorMe is the one running on the smartphone, since it performs most of the processing that allows online monitoring:

- Data collection from the smartphone and MARG module sensors
- Information extraction from the sensor data
- Classification of the current activity given the extracted information
- Speed estimation when the user is walking/running
- Video recording with variable frame rate according to the user speed
- Transmission of the current recognized activity and the video being recorded to a remote device, via a server.

1.3 Contributions

The main contribution of this dissertation is the development of an online monitoring system, based on motion sensors and a camera, using two mobile devices. The most important part of this system is an Android application, which includes an activity recognition algorithm. Another important contribution is the frame rate control of a video being recorded, according to the user speed.

The developed activity recognition algorithm allows the identification of different types of activities, including static, dynamic and instantaneous activities. The algorithm was implemented after analyzing a set of sensor data collected from various subjects, as described in detail in this dissertation. The main characteristics of the recognition algorithm are its low computational cost and the use of only one type of sensor (accelerometer). It is also possible to use just one device (a smartphone placed at the chest), if the distinction between the sitting and standing postures is not important.

The video frame rate control allows further savings in processing requirements and also a decrease of the necessary transmission bandwidth. For the purpose of frame rate control, the implemented system estimates user speed while walking and running. The speed estimator was developed after the study of data collected in a real track and field environment. The validation of the resulting algorithm was also performed with data obtained in the same experiment. The speed estimator needs only a feature already computed for activity recognition, and some simple additional calculations. This represents a good alternative to the use of the GPS for obtaining speed, since the GPS would use even more phone resources.

Another contribution consists in the integration of all developed algorithms into a system which allows transmission of all the resulting information (recognized activity plus images) to a server through a wireless connection. Furthermore, this information can be accessed online by a remote device connected to the server using any type of connection. The operation of this system was demonstrated in a real situation.

A paper describing the MonitorMe system was submitted to the 8th IEEE International Conference on Wirless and Mobile Computing, Networking and Communications (WiMob 2012), to be held in Barcelona, October 8-10, 2012.

1.4 Dissertation Structure

This dissertation is divided into 5 chapters, excluding this one. **Chapter 2** provides some background on the subjects studied in this dissertation: activity recognition, speed estimation and video monitoring. For the first two areas, an overview of the state of the art is also presented.

The proposed system, MonitorMe, is detailed in **Chapter 3**. This chapter begins by describing the requirements of a monitoring system based on sensors. The specific case of the MonitorMe is then explained, including its additional feature of video recording with adaptable frame rate. This chapter describes also the MonitorMe architecture, including the specifications of the used sensor devices. Details on the programming environment and on the MonitorMe implementation are also presented.

The development of the MonitorMe activity recognition and speed estimation algorithms is described in **Chapter 4**. The chapter starts by detailing the methods used to collect the experimental data. Then the activity recognition algorithm is presented, followed by details on the data analysis performed in order to develop this algorithm. The algorithm used for controlling the video frame rate is also presented, with special emphasis on the speed estimator developed from the sensor data analysis. In the final part of this chapter, the results of the algorithms evaluation are presented, including a discussion of the achieved values and a comparison with some data from the literature.

Chapter 5 includes a demonstration of the MonitorMe system in a real situation, with online recognition of activities and video frame rate control. The relevant information was saved during the demonstration, allowing the confirmation of the activity recognition and of the frame rate variation according to the user activity/speed.

Finally, **Chapter 6** summarizes the contents and presents some of the main conclusions of this dissertation. In this chapter, some possible future research topics are also described.

2 Background and Related Work

The monitoring of a person can be achieved in different ways. In this dissertation, activity recognition, video recording and speed estimation are used together for online monitoring. This chapter provides some background on the most used methods/techniques and describes some of the work done in each of these areas.

2.1 Activity Recognition

Activity recognition is usually achieved by performing the sequence of main steps shown in Figure 1. Different types of data can be used to recognize activities, but the most commonly used are the data obtained from one or more sensors. From these data it is possible to extract features, which are the input of a classifier that is able to identify a given set of activities. These activities are generally simple day-to-day activities, such as standing, sitting, lying and walking. A brief background on sensors, features and classifiers is presented next. This section ends with a summary of related work in the area of activity recognition.

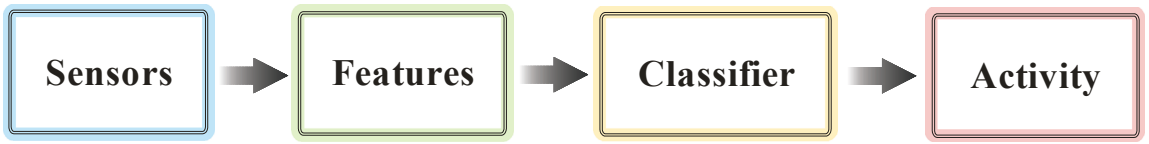


Figure 1. Sequence of steps usually needed for activity recognition.

Accelerometer and Gyroscope Sensors

In a very broad way, a sensor can be defined as a device that is able to convert a physical stimulus (e.g. heat, light, sound, pressure, magnetic field, a specific motion) into an signal that can be measured [6]. In the recent years an increasing number of micro sensors, using MEMS (Micro-electromechanical Systems) technology, have been integrated into portable devices (e.g. smartphones, wearable devices). This contributed to the use of these sensors with the aim of recognizing different activities. Smartphones are carried on a daily basis by most people, making the combination of smartphones and sensors suitable for daily monitoring of activity.

Some sensors have already been used in mobile devices for quite some time, including audio sensors (e.g. microphones), vision sensors (e.g. cameras) and GPS sensors. Other types of sensors are also being integrated into the most recent devices [7]. In the particular case of motion sensors, they have been used mainly for gaming and music player control, and for automatic screen rotation. Motion sensors have also been used intensively in research for activity recognition [8-11] and also for speed estimation [8-12].

In this dissertation the accelerometer and gyroscope (motion sensors) were studied for this purpose. The gyroscope measures the device's rate of rotation, i.e. the device's angular speed around each

axis (in rad/s). There are three physical axes (x, y, z) and the sensors that have an associated coordinate system can be uni-, bi- or tri-axial.

The accelerometer deserves some additional explanation. This device measures the specific force, defined as the non-gravitational force per unit mass. This quantity, also called g-force, is measured in meters/second^2 (m/s^2), which are the units of acceleration in the International System of Units (SI). So the specific force is not really a force, but a type of acceleration. However, this is not the coordinate acceleration used in many situations, but the proper acceleration. The latter is the acceleration relative to the free-fall reference frame (inertial reference frame).

For example, if an accelerometer is resting on a horizontal surface with its z-axis facing up, it will measure a value of $+g$ on this axis (g = 9.80665 m/s^2). This happens because, although it is not accelerating relative to a frame associated with the surface (zero coordinate acceleration), it is accelerating upwards relative to the freely-falling frame. Alternatively we can say that when resting on the horizontal surface, the accelerometer measures the g-force exerted by the surface.

On the other hand, the accelerometer will show a zero value when in free-fall toward the center of the earth, since although the phone is accelerating relative to a frame associated to the earth's surface, it is at rest in the free-fall frame of reference (in which objects are weightless). From another point of view, we can say that the specific force is zero for free-falling objects, because gravity alone does not produce g-forces.

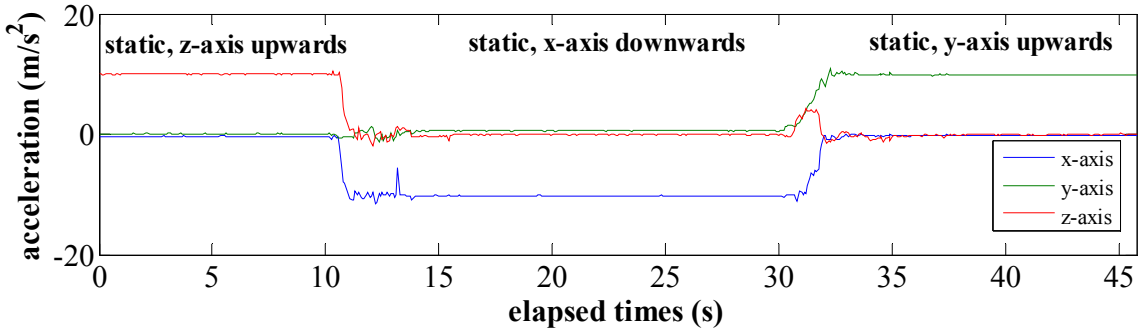
The examples above show that proper acceleration can be rather different from coordinate acceleration. However, since it will be the only type used in this dissertation, the proper acceleration will be denoted simply as acceleration in the remainder of this document.

To further illustrate the accelerometer and gyroscope properties, some data were collected from the smartphone used in this dissertation (Samsung Galaxy S II). Figure 2 shows the values of both sensors for different static positions of the smartphone. Firstly, the phone was kept on a flat surface with the screen facing upwards (positive z-axis parallel to the force of gravity but in the opposite direction). The values obtained are shown in Figure 2 (a), where it can be seen that the reading on the z-axis is around 10 m/s^2 (it should be 9.8 m/s^2 if the z-axis was exactly parallel to the gravitational field). On the other hand, the values on the x- and y-axis are approximately zero. This is in accordance with the explanation given above, since as the accelerometer is resting on a horizontal surface, it measures the g-force exerted by the surface, which in this case is $+g$ (direction of the positive z-axis).

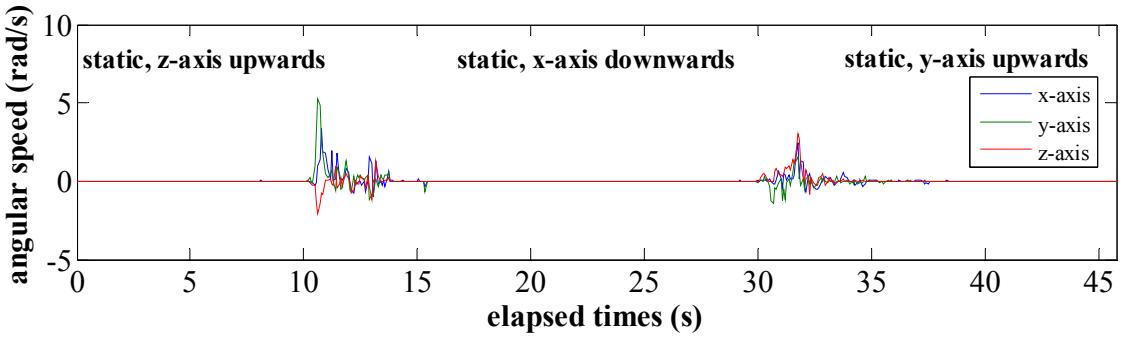
After a few seconds the smartphone position was changed so that its volume control was on top (positive x-axis parallel and in the same direction as the force of gravity). In this case, the g-force exerted by the surface is directed towards the negative x-axis so that the acceleration value becomes approximately $-g$ on the x-axis and zero on the other two axes, as seen in Figure 2 (a). The random variations between the two static positions of the smartphone are due to the movement of the smartphone between these two positions.

Finally, the smartphone was positioned so that its headphones' input was on top (positive y-axis parallel to the force of gravity but in the opposite direction). As expected, now the acceleration value is approximately $+g$ on the y-axis and zero on the two other axes.

As shown in Figure 2 (b), the behavior of the gyroscope is simpler: whenever the smartphone is static the reading on all three axes is approximately zero; only when the phone is moved from one static position to the next position the angular speed takes random values that depend on the rotation around the corresponding axis.



(a)



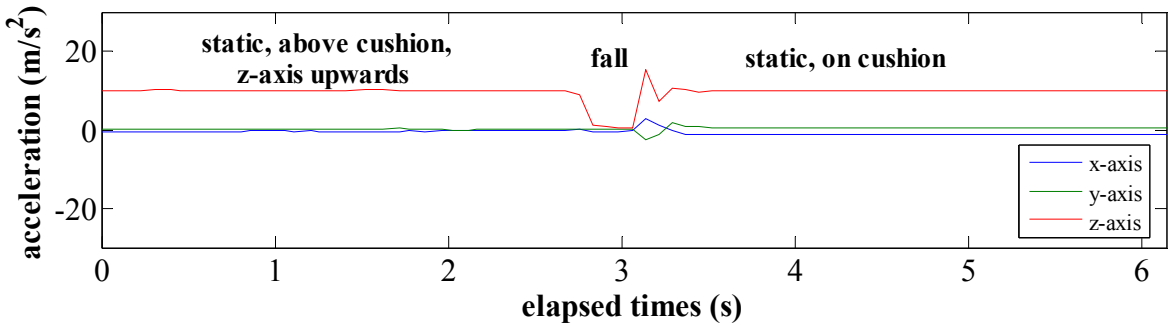
(b)

Figure 2. Values for the phone accelerometer (a) and gyroscope (b) on the three axes, when changing between three static positions.

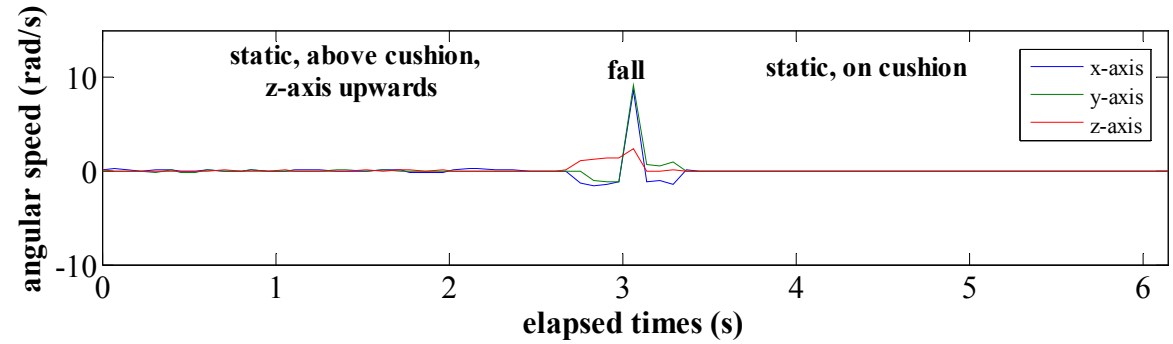
Figure 3 illustrates the sensors dynamic behavior. In the beginning the smartphone was held approximately 60 cm above a cushion, with the screen facing upwards (positive z-axis parallel to the force of gravity but in the opposite direction). As it can be seen in Figure 3 (a), the acceleration on the z-axis is approximately +*g* and zero on the x- and y-axis. This situation is similar to the one shown above (phone on a flat horizontal surface). But now the values change around an average value due to the fact that, while holding the smartphone, the hands were not completely still. For the same reason the gyroscope values, Figure 3 (b), vary also around an average value of approximately zero.

After a few seconds in the static position, the smartphone was dropped and fell on the cushion. This corresponds to the fast changes in the central part of Figure 3. The z-axis value for the accelerometer is especially interesting: it changes from a value of approximately +*g* to a value of zero. The latter corresponds to the time interval when the phone was in free-fall: as explained above, in this situation the g-force, or proper acceleration, is zero. After this zero value the acceleration increases instantaneously to a value greater than *g* (≈15 m/s^2). This high value corresponds to a large g-force exerted by the cushion, to decelerate the phone in a short time interval. After this, the z-axis acceleration returns to its initial value of approximately +g. On the

other hand, the acceleration on the two other axes started by having approximately a zero value, but when on the cushion, the phone's y-axis acceleration is greater than zero and the x-axis acceleration is smaller than zero. This behavior is due to the fact that after the fall the phone took a position that was not exactly horizontal.



(a)



(b)

Figure 3. Values for the phone accelerometer (a) and gyroscope (b) on the three axes, when dropping from a static position.

Features

From the data collected from sensors it is generally possible to obtain further information. In the activity recognition area, this information usually corresponds to one or more features extracted from the sensor data, for time windows of finite and constant size, which varies a lot from work to work. It is also common to use sliding windows, i.e. to have some overlap between consecutive windows (normally of 50%).

Features can be classified according to their domain: time, frequency or discrete [13]. Time-domain features can be obtained using mathematical or statistical functions, such as the mean, variance, root mean square (RMS), minimum and correlation, but also using other type of functions (e.g. zero-crossing). Some examples of frequency-domain features are the DC component, energy and entropy. Belonging to the discrete-domain, we can have symbolic string representations, such as the Levenshtein edit distance. These and other features are described in detail in [13], where they also show that time-domain features usually have lower computational cost, when comparing to frequency-domain or discrete-domain features. This is due to the fact that the time-domain features are computed directly over the sensor data, while to obtain the frequency- and discrete-domain features a transformation from the time-domain to the desired domain is needed.

Classifiers

Considering a set of *N* classes, a classifier is an algorithm that is able to identify a given observation as belonging to one of the existing classes [14]. A classifier is normally based on a set of observations whose class association is known. In the sensor-based activity recognition context, a classifier is used to identify different activities given a set of features extracted from sensor data. Some of the most popular techniques used in classifiers include the following: k-nearest Neighbor (*k*-NN), Nearest Mean (NM), Neural Networks (NN), Hidden Markov Models (HMM), logistic regression, decision trees and threshold-based. These and other techniques are described in [15].

The threshold-based technique consists in developing an algorithm that classifies activities according to one or more thresholds. These thresholds are defined during the analysis of extracted features and their value ranges for each of the considered activity. This was the technique used in this dissertation to construct a classifier for activity recognition, since it enables more control in the choice of the most adequate recognition strategy for each activity, while achieving similar or better accuracies when compared with some classifiers constructed in a more automated way. The latter are generally obtained automatically when the generating technique is provided with an input of features. This technique can be one of the listed above (except the threshold-based), which are not further detailed in this dissertation.

Related Work

In recent years there has been intense research in the activity recognition area. Studies related to activity recognition use mainly accelerometers [8, 9, 13, 15-24]. However, combinations of different types of sensors have also been used, such as an accelerometer together with an orientation sensor and a magnetic field sensor [25]. Recently the gyroscope has also been used either alone or combined with other sensors [26-28]. Wearable devices have been widely used in many studies [8, 9, 15, 16, 20, 21, 23, 24, 26-28]. Nowadays smartphones[1] include many embedded sensors, which make these devices increasingly popular in activity recognition [17-19, 22, 25].

Table 1 presents a state of the art summary of activity recognition research, with emphasis on the most relevant studies for this dissertation. For each reference, the table lists the devices, sensors and locations used, the activities recognized, the number of subjects that participated, the sampling rate and window size used and the features extracted. Some of the techniques used for obtaining the classifiers are also listed, as well as the corresponding achieved accuracies. The word accuracy is used here for simplicity, since its definition may vary from work to work. More details concerning accuracy for each reference (if defined) can be found below.

As already mentioned above, wearable sensors have been often used in activity recognition. Some studies used these sensors across multiple body locations, such as [15] where they used sensor data collected by Bao and Intille [21] from 13 users, each using bi-axial accelerometers placed at 5 different body locations (see Table 1). The main goal of [15] was the development of a pedestrian navigation system, so they focused on activities where the legs have an important role (e.g. sitting,

[1] Although smartphones can also be considered wearable devices, in this dissertation the term wearable devices is used for all devices in this category except for smartphones and Wii remotes.

standing, walking, running). For each sensor location and each axis the following features were extracted: DC component, energy, entropy and correlation coefficients between each pair of signals. Classifiers using several techniques, including the ones listed in Table 1 for [15], were trained with the referred features (separately for each subject). The evaluation of these classifiers showed that the best results (> 98%) were achieved for the classifiers based on NM, k-NN and cHMM (continuous emissions HMM) with 2 training phases. The exact meaning of accuracy is not defined in this publication: it is described as the "the aggregate classification accuracy", obtained from the "aggregated confusion matrix that added the classification outcomes for all subjects".

In contrast, there are studies that focused only on 1 or 2 body locations, being the most popular the trunk/chest, the waist and/or the leg/thigh. In [16] a single wearable tri-axial accelerometer was placed at the chest to recognize 5 different activities (walking, stair climbing, standing, talking with people, and working at computer). They applied a filter to the collected sensor data to obtain the low and high frequencies components. The features listed in Table 1 were extracted from the signals with and without filtering, considering the values on three axes of the accelerometer and the acceleration magnitude (square root of the sum of the value on each axis squared). The skewness feature is a measure of the degree of asymmetry of a distribution around its center point and the kurtosis measures if the distribution is peaked or flat relative to the normal distribution[2]. The correlation was computed for each pair of accelerometer axes. The energy was obtained from the coefficients of a wavelet transform. The root mean square (RMS) of speed was also calculated, where the speed was obtained by integrating the acceleration. The Minmax sums correspond to the sum of all the samples in a window sorted in crescent order, with all the samples in the same window sorted in decreasing order. The features extracted from the training set were used as the input of a random forest, which associated a measure of importance to each feature. The 20 most meaningful features were chosen to train the classifiers based on a decision tree, a bagging[3] of 10 decision trees, AdaBoost[4] (Adaptive Boosting) using decision trees and a random forest of 10 decision trees. The selected features included mean values, standard deviations and mean values of Minmax sums of acceleration, and root mean squares (RMS) of speed. Accuracies greater than 90% were observed for all 4 classifiers.

[2] For example, data sets with a low kurtosis, such as the uniform distribution, tend to have a flat top around the mean. In contrast, data sets with a low kurtosis tend to have sharp peaks near the mean.
[3] Bagging (bootstrap aggregating) is a method for combining decision trees or other classifiers.
[4] The AdaBoost technique is another method for combining low-accuracy classifiers to create a high-accuracy classifier.

Table 1. State of the art summary of activity recognition research

Reference	Devices, sensors & locations	Activities	Subjects, sampling rate & window size	Features	Techniques used in classifiers	Accuracies (%)
[15]	5 wearable devices each with a bi-axial accelerometer, at hip, wrist, arm, ankle and thigh	– Sitting – Lying – Standing – Walking – Stair climbing – Running – Cycling	13 subjects, 76.25 Hz & 6.7 seconds (50% overlap)	– DC component – Energy – Entropy – Correlation coefficients	– Logistic regression – NM – *k*-NN – Artificial NN (ANN) – Binary decision tree (C4.5 algorithm) – cHMM	– [92, 99]
[16]	Wearable device with a tri-axial accelerometer, at chest	– Walking – Stair climbing – Standing – Talking with people – Working at computer	14 subjects, 52 Hz & 1 second (50% overlap)	– Mean – Standard deviation – Skewness – Kurtosis – Correlation – Energy – RMS of speed – Mean of Minmax sums	– Decision Tree – Bagging of 10 decision trees – AdaBoost (Adaptive Boosting) using decision trees – Random forest of 10 decision trees	– [90, 94]
[22]	Mobile phone with a tri-axial accelerometer, in front pants leg pocket	– Walking – Jogging – Stair climbing – Sitting – Standing	29 subjects, 20 Hz & 10 seconds	– Mean – Standard deviation – Mean absolute difference – Mean resultant acceleration – Time between peaks – Binned distribution	– Decision tree (C4.5 algorithm) – Logistic regression – Multilayer NN (MNN)	– [56, 97[for the decision tree – [12, 98] for logistic regression – [44, 98] for MNN

Reference	Devices, sensors & locations	Activities	Subjects, sampling rate & window size	Features	Techniques used in classifiers	Accuracies (%)
[25]	Mobile phone with a tri-axial accelerometer, orientation and magnetic field sensors, in front pants pocket	– Walking – Running – Stair climbing	2 subjects, 64 Hz & 2 seconds (50% overlap)	– Mean – Variance – Range – Energy	– Decision tree (C4.5 algorithm)	– [58, 82[
[13]	Wii Remote with a tri-axial accelerometer, in right pants pocket	– Walking – Running – Jumping	Nr. of subjects not specified, 100 Hz & 2.56 seconds (50% overlap)	– Various time-domain – Various frequency-domain – Various discrete-domain	– Threshold-based	– [41, 81[(time-domain) – [58, 80[(frequency-domain) – [32, 53[(discrete-domain)
[23]	2 wearable devices each with a uni-axial accelerometer, at chest and thigh	– Sitting – Standing – Lying – Locomotion	5 subjects, 10 Hz & 10 seconds	– Median – Mean absolute deviation	– Threshold-based	– [73, 93]
[27]	3 wearable devices each with a tri-axial accelerometer and a tri-axial gyroscope, at chest and thighs	– Sitting – Standing – Lying – Locomotion – Transition	10 subjects, 50 Hz & 1 second	– Standard deviation – Number of peaks – Power spectral density	– Threshold-based	– [96, 100]
[20]	2 wearable devices each with a tri-axial accelerometer, at trunk and thigh	– 8 types of falls – 7 ways of sitting and lying – Walking	10 + 10 subjects, 1 kHz & window size not specified	– Root sum of squares	– Threshold-based	– [67, 100] for falls – [23, 100] for other activities
[28]	Wearable device with a bi-axial gyroscope, at trunk	– 8 types of falls – 7 ways of sitting and lying – Walking	10 + 10 subjects, 1 kHz & window size not specified	– Root sum of squares	– Threshold-based	– 100

As mentioned before, mobile phones are increasingly being used in activity recognition studies, such as in [22] where an Android smartphone including a tri-axial accelerometer was placed in the front pants leg pocket to identify 6 different activities: sitting, standing, ascending and descending stairs, walking, and jogging. Sensor data was collected from 29 users while carrying out these activities. Two different window sizes were studied, 10 and 20 seconds, with the best results for 10-second windows. For each window, the mean, standard deviation, average absolute difference, and time between peaks were computed for each axis. The average absolute difference is the average of the absolute difference between each sample value and the mean value of the corresponding window. Another feature computed was the mean of the resultant acceleration (square root of the sum of the value on each axis squared). The binned distribution was also obtained, for each window, by dividing the range of values on each axis into 10 bins with equal sizes. Each bin contained the fraction of the measured values that fell in that bin. This set of features was used to train 3 classifiers: decision tree (using J48, a Java implementation of C4.5 algorithm[5]), logistic regression and multilayer neural networks (MNN). Overall, the MNN classifier had the best accuracy (92%), where accuracy for each activity is defined as the percentage of correctly classified windows, when considering the total windows for a given activity.

Even though the accelerometer is the most used sensor for activity recognition, others sensors have been used alone or in combination. For example, in [25] a 3-axial accelerometer, an orientation sensor and a magnetic field sensor, available in a smartphone, were explored. These sensors were used together to recognize the following activities: walking, running, climbing stairs, and descending stairs. Sensor data were collected from the mobile phone while being carried by 2 subjects in the front pants pocket. The acceleration data were normalized, using the orientation and magnetic field data, so that the acceleration values were independent of the device orientation. The mean, variance, range (difference between the maximum and minimum values) and energy were extracted from the normalized data belonging to the training set. These features were used to train a decision tree (using a Java implementation based on the C4.5 algorithm – jaDTi), considering windows of 1, 2 and 3 seconds with 50% overlap. The classifier evaluation showed that for walking and running the best results were achieved for 1-second windows, while for climbing and descending stairs the best accuracies were achieved for 2-second windows. Nevertheless, the best overall accuracy was obtained for 2-second window. This was the window size used for a final study, where a constant sample rate of 64 Hz was set, obtaining accuracies of 78% and 82% for walking and running respectively, and near 60% for ascending and descending stairs.

In [13] a study of mobile devices suitability for implementation of activity recognition was performed, taking into account the computational cost as well as the storage and memory operations necessary to compute each feature. The main conclusion was that most of the considered time-domain features are suitable for implementation in mobile devices, while for the frequency-domain and discrete-domain cases just one the considered features is suitable. An experimental study was also performed in [13], using a Nintendo Wii Remote with a tri-axial accelerometer, in order to verify the suitability of the features for the recognition of three dynamic activities: walking, running and jumping. The data from the training set was divided into 2.56-seconds (256

[5] C4.5 is an algorithm used to generate decision trees. J48 is an open source Java implementation of the C4.5 algorithm.

samples) windows, which was the basic unit used for classification. For each sample, the norm (square root of the sum of the value on each axis squared) was computed. For each window, the various time-, frequency- and discrete-domain features were extracted from the norm values (with some possible exceptions, such as the sample differences, where the absolute difference between signals is computed in a pairwise arrangement of samples from the 3 axes). For each feature, two thresholds were defined to distinguish between the 3 considered activities. The optimum values of these thresholds were obtained by maximizing the accuracy when classifying the training set. The developed classifier was then evaluated using the test set. Accuracy is defined as the percentage of windows correctly classified. Considering the 3 activities, the best results were achieved for the minimum value (over a window) and the sample differences, both belonging to the time-domain, with accuracies of 80% and 81% respectively. In the frequency-domain, the best accuracies were obtained for the energy and the fast Fourier transform (FFT) coefficients sum (from 0.5 Hz to 3 Hz), with accuracies of 77% and 80% respectively. The discrete-domain features analyzed had much lower accuracies (below 53%).

Regarding threshold-based algorithms, some studies [20, 23, 24, 27, 28] used this technique and a configuration with sensors placed at the chest/trunk and/or at the thigh(s). In [23] two uni-axial accelerometers were placed at the chest and thigh to identify static activities (lying, sitting and standing) and dynamic activities (locomotion). A threshold-based algorithm was developed using the data collected from 5 subjects with a sample rate of 10 Hz. For each time interval of 1 second, the median of the low-pass filtered acceleration in both locations (chest and thigh) was computed. The sum of the mean absolute deviation for both locations was also calculated, where the mean absolute deviation is the average of the absolute difference between each sample value and the mean value of a given 1-second time interval. Then, for each window of 10 seconds, the average value of these features was obtained. The developed algorithm achieved accuracies of 73% for standing, 91% for lying and locomotion and 93% for sitting, where the accuracy for each activity was defined as the percentage of time in which the recognition was correct.

A combination of accelerometers and gyroscopes was used in [27]. More specifically, they used 3 wearable devices, each with a tri-axial accelerometer and a tri-axial gyroscope. These devices were placed at the chest and both thighs, and they sent the sensor data to a smartphone though a Bluetooth connection. Part of the data collected from 10 subjects was used to develop a threshold-based algorithm that recognizes sitting, standing, lying and locomotion activities, and transitions between these activities. The angle of each body part (chest and both tights) relative to the vertical axis is calculated from the accelerometer and gyroscope data, using three types of filters and numerical integration. The features extracted from the angle values over 1-second windows were the following: mean, standard deviation, number of peaks, maximum difference relative to the mean and power spectral density[6]. Using a validation similar to the one used in [23] (described above), accuracies of 96% and 99% were observed for standing and sitting respectively, and of 100% for the other considered activities and for transitions.

Fall detection is also an important aspect of the activity recognition algorithm developed in this dissertation. Some studies, such as [20, 26, 28], focused on fall detection. In these situations it is

[6] The power spectral density characterizes a random signal in the frequency domain. It is useful to determine the frequency content of the signal.

common to analyze also other activities that can be confused with falls (such as transitions from standing to sitting or lying, and the walking activity). In [20] two tri-axial accelerometers were placed at the trunk and thigh to detect falls in elderly people. In a first experiment, 10 young subjects simulated 8 different fall types. Another experience consisted in having 10 elderly subjects performing activities of daily living (ADL) that can be incorrectly recognized as falls, such as sitting down and standing up from different types of seats, lying down and standing up from a bed, and walking. The collected signals were low-pass filtered and then the root sum of squares (square root of the sum of the value on each axis squared) was computed for both locations. For each location, an upper and a lower fall threshold were defined, considering the positive peaks and the negative peaks, respectively. They verified that only the upper fall threshold associated to the trunk achieved 100% accuracy in fall detection. Also, all ADLs were correctly recognized as non-falls using this threshold.

In [28] a study similar to [20] is made, using a bi-axial gyroscope placed at the trunk. The rest of the setup is the same as [20], concerning the activities, number of subjects, sample rate and feature computed. In addition to the angular speed, the angular acceleration (derivative of angular speed) and the change in trunk angle (integration of angular speed) were obtained for each of the two axes. The resultant values for angular speed, angular acceleration and change in angle were then obtained by evaluating the root sum of squares. Three thresholds were defined for the resultant angular speed, the maximum value of change in angle and the maximum value the angular acceleration. Using these 3 thresholds, falls were always correctly identified (100% accuracy) and all the ADLs were correctly recognized as non-falls.

2.2 Speed Estimation

Speed is a scalar measure of the rate of change of a body position. The average speed is the distance travelled by the body divided by the elapsed time. The instantaneous speed is the limit of the average speed as the time interval duration approaches zero. The SI units of speed are meter per second (m/s).

One of the objectives of this dissertation is to estimate the user speed while walking or running, for control of the frame rate of a video recorded by the smartphone camera during online monitoring. Table 2 summarizes the methods used in [8-11, 24] for estimating the subject speed based on sensor data. For each reference, the table lists the devices, sensors and locations used, the speed range, the number of subjects that participated, the sampling rate and window size used, the features/methods used for estimating speed and the corresponding evaluation results. When some information is not included in the table, it is due to the lack of its specification in the corresponding study.

In [24] they observed that the mean value of the duration of a signal cycle and the standard deviation of the acceleration values, obtained from the data collected from an uni-axial accelerometer placed at the thigh, can possibly be related with the user speed in dynamic activities. However, the actual speeds associated with the data were not measured and further studies were not presented, since this was not the objective of this paper.

Table 2. State of the art summary of speed estimation research

Reference	Devices, sensors & locations	Speed range (m/s)	Subjects, sampling & window size	Features/Methods	Evaluation results
[24]	1 wearable device with a uni-axial accelerometer, at thigh	—	5 subjects, 100 Hz	– Mean duration of signal cycles – Standard deviation of acceleration values	—
[9]	1 wearable device with a tri-axial accelerometer, at chest	—	10 subjects, 20 Hz & 10 seconds (50% overlap)	– Distance covered (obtained from stride length and number of zero crossings) in a window – Window duration	– Error rate of 6.5% for step counter (using the number of zero crossings)
[10]	1 wearable device with a tri-axial accelerometer, at chest	[1.31, 4.76]	17 subjects, 200 Hz	– Range value of strides, and user weight and height for stride length estimation (by 3 neural networks) – Stride duration	– Coefficient of determination of 0.975 – Mean squared error (MSE) of 0.0225 $(m/s)^2$
[8]	3 wearable devices each with a bi-axial accelerometer	[1.12, 6.48[	1 subject, 100 Hz & 1 second	– RMS of resultant acceleration	—
[11]	2 wearable devices each with bi-axial accelerometer, at both thighs	[0.28, 3.61]	5 subjects, 25 Hz & 2 seconds	– Average of the summations of resultant acceleration for each thigh – 3rd-order polynomial model	– MSE of 0.1358 $(m/s)^2$

A tri-axial accelerometer placed at the chest was used in [9] to extract exercise related information, including the stride length, step count, distance, speed and energy expenditure. If the current activity being carried out by the subject was walking, the speed was obtained by using the distance covered during a 10-second window and the window duration. The distance was calculated from the stride length and the step count for the considered window. The stride length was estimated for each user based on the age, sex and height of the subject. The number of steps was obtained from the number of zero crossings. The validation of the step counter led to an error rate of 6.5% for walking, considering the estimated and actual total number of steps.

A recent review of methods for speed estimation based on sensors is made in [12]. Most of the described studies explored only lower speeds, but others such as [10, 11] also analyzed higher speeds. In [10] data was collected on a treadmill, from a tri-axial accelerometer placed at the chest. These data were segmented into strides. For each stride and each of the 3 axes, the difference between the maximum and the minimum values was computed. These features, together with the subject's height and weight were used as an input of three neural networks, one for differentiating

walking from running, and the other two for estimating stride length for walking and for running, respectively. The speed was calculated using the estimated stride length and the corresponding stride duration. The latter was measured during the initial segmentation of the acceleration signal into strides. An evaluation of this speed estimator led to a coefficient of determination of 0.975 when considering measured and estimated speeds, and a mean squared error (MSE) of 0.0225 $(m/s)^2$, for measured speeds in the range of [1.31, 4.76] m/s. An additional evaluation was made, with a user running 10 km on a track and field environment. An average estimated speed of 4.76 m/s was obtained for an actual average speed of 5.27 m/s, which results in a relative error of 9.59%.

Another way of estimating speed, instead of computing stride/cycle length and duration, is to use directly a feature extracted from the sensor data. This method was used in [8], where they investigated the possibility of determining the speed while performing the following activities: walking, jogging, running and sprinting. Data were collected from bi-axial accelerometers, placed at the side of the chest and at the side of the right thigh, while a user performed the referred activities. For each location (chest and thigh), the RMS of the resultant acceleration was computed over 1-second windows. The average value of all windows, for each activity and location combination, was obtained. Just one subject was used for this study and no validation tests were performed with alternative users.

In [11] bi-axial accelerometers placed at both thighs were used to estimate speed from acceleration values. The sum of the resultant accelerations (square root of the sum of the value on each axis squared), associated with the left and right thighs, was averaged over each 2-second window. Then, for each speed and subject combination (five subjects were used), the average of all windows results was calculated. Finally, the most adequate third-order polynomial model was obtained from all speed-subject data. An evaluation of this speed estimator led to an overall MSE of 0.1358 $(m/s)^2$, for speeds in the range of [0.28, 3.61] m/s.

2.3 Video Monitoring

A video is a sequence of still images that represent scenes in motion, which nowadays are usually captured by a camera with an electronic image sensor (digital camera). The number of still pictures per unit of time (or frames per second) corresponds to the frame rate of the video. With the aim of improving the compression rate, a video can have a variable frame rate for example by reducing the number of frames or by dropping frames during intervals of time where there is a large amount of static images. Variable frame rate is used in [29] for real-time video communication adapted to deaf users using mobile phones, in order to save system resources. Regarding the use of mobile phones, these devices are suitable for image capture, since they are carried on a daily basis, and most of the recent models include a camera.

A video can be recorded and stored for a later view, but it can also be streamed live. A live video stream or broadcast is the online transmission of video content normally over the Internet in a continuous stream of data, which can be displayed to one or more users. As it will be explained later, in this dissertation a smartphone is used to capture still images, which are sent sequentially to a remote user. This means that it was necessary to deal with images formats, instead of video

formats. The most used image file formats in digital cameras are the following [30]: JPEG/JFIF (Joint Photographic Experts Group/JPEG File Interchange Format), TIFF (Tagged Image File Format) and raw file formats. If it is important to store or send images in a more efficient way, image compression can be used to reduce the size in bytes while still having an acceptable level of quality.

Video monitoring can be defined as the act of monitoring by recording a video of the place/person to be supervised. This monitoring can be made remotely by someone who watches a video that was or is being recorded by a camera placed at the location of the object/person to be monitored. This type of monitoring is used in the healthcare area to supervise patients at the hospital [31, 32] or elderly people in their own homes [33, 34]. Video monitoring can also be useful in some professions, such as in the security and emergency service areas. The capture of a video can also be used for leisure, for example in extreme sports or to log the day-to-day activities of a person. In all these scenarios images of the surrounding environment can be captured using a mobile phone camera, a special head mounted camera [35] or glasses with an integrated camera [36, 37].

3 MonitorMe System

This chapter presents MonitorMe, a system for the online monitoring of a person's day-to-day activities. The chapter is divided into 3 sections. The first section describes the design of a monitoring system based on sensors. This is followed by details on the specific case of the MonitorMe design, including some non-functional requirements. The second section presents the architecture of the implemented system, including the most relevant specifications of the used sensor devices. The final section of this chapter describes the programming environment and provides details on the implementation of the MonitorMe smartphone application.

3.1 Monitoring System

One of the main objectives of this dissertation is the implementation of a system that allows the online monitoring of a given person, by recognizing different activities based on sensor data. This scenario includes also the user of a remote monitoring device. The system should be able perform the following tasks:

- Collect data from sensors worn by a subject
- Extract features from the obtained sensor data
- Recognize the activity being carried out by the user, based on the features extracted
- Send the information of the current activity to a remote monitoring device
- Present the activity information online to the user of the remote device.

These tasks are presented as use cases in Figure 4, where the main use case ("Recognize activities") is sub-divided into the tasks shown in a red background in Figure 5.

MonitorMe

In the case of the MonitorMe system, the user being monitored wears a smartphone that includes an accelerometer and a gyroscope, and an in-house built module based on MARG (Magnetic, Angular Rate and Gravity) sensors [5]. The sensor data is gathered and processed by the smartphone, which in addition to recognizing activities, simultaneously captures images using its camera. MonitorMe also estimates the speed of the user using features extracted from the sensor data, which is used to control the image frame rate. The activity information and the captured images are sent to a server, which relays them to the remote monitoring device. Figure 6 shows the tasks that are additionally performed by the MonitorMe system (in green) and their relationship with the basic tasks of a sensor-based monitoring system that performs activity recognition. More details on these tasks are included in the Section 3.3.2, where the most important aspects of the smartphone application implementation are described.

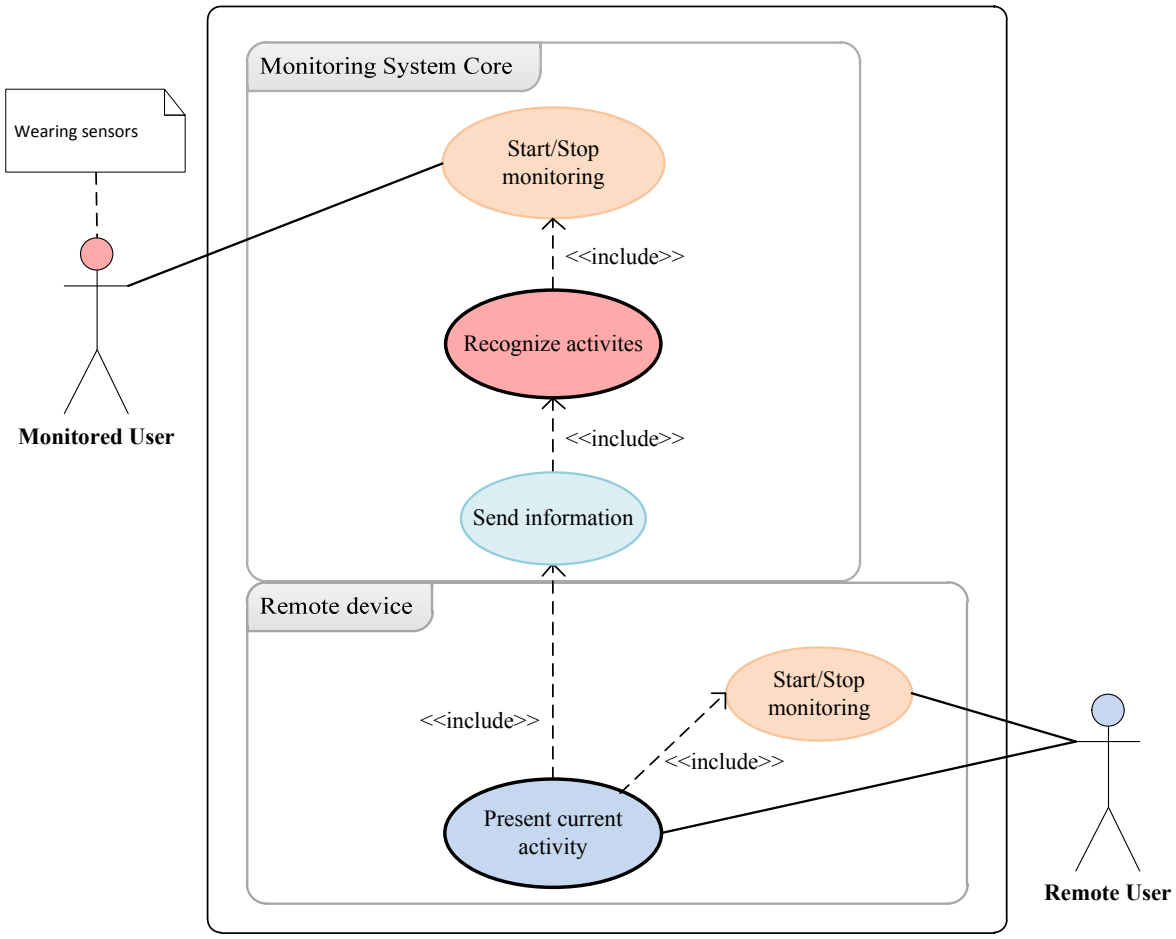


Figure 4. Use case diagram of an online monitoring system that performs activity recognition.

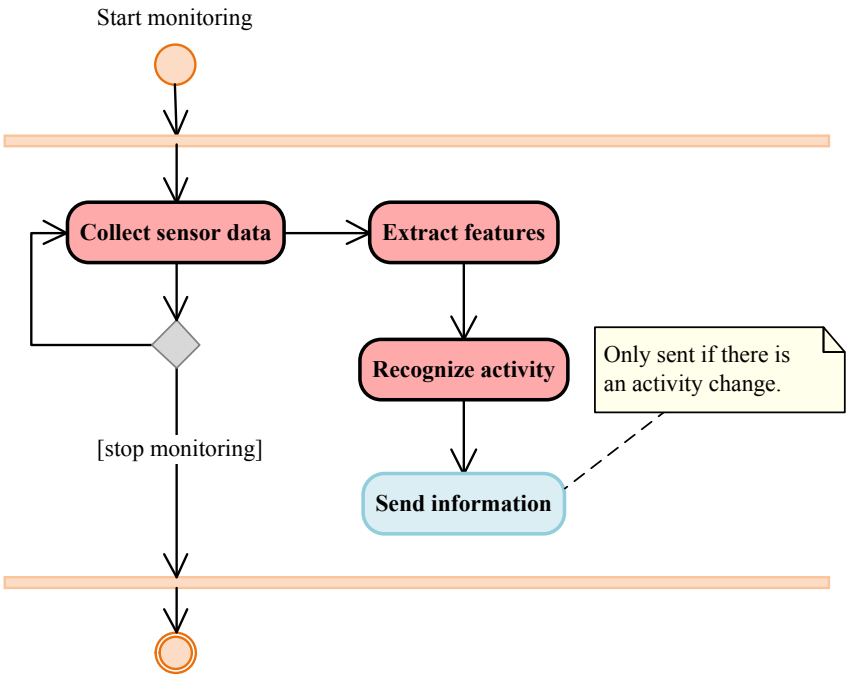


Figure 5. Activity diagram that includes the various tasks necessary for online activity recognition.

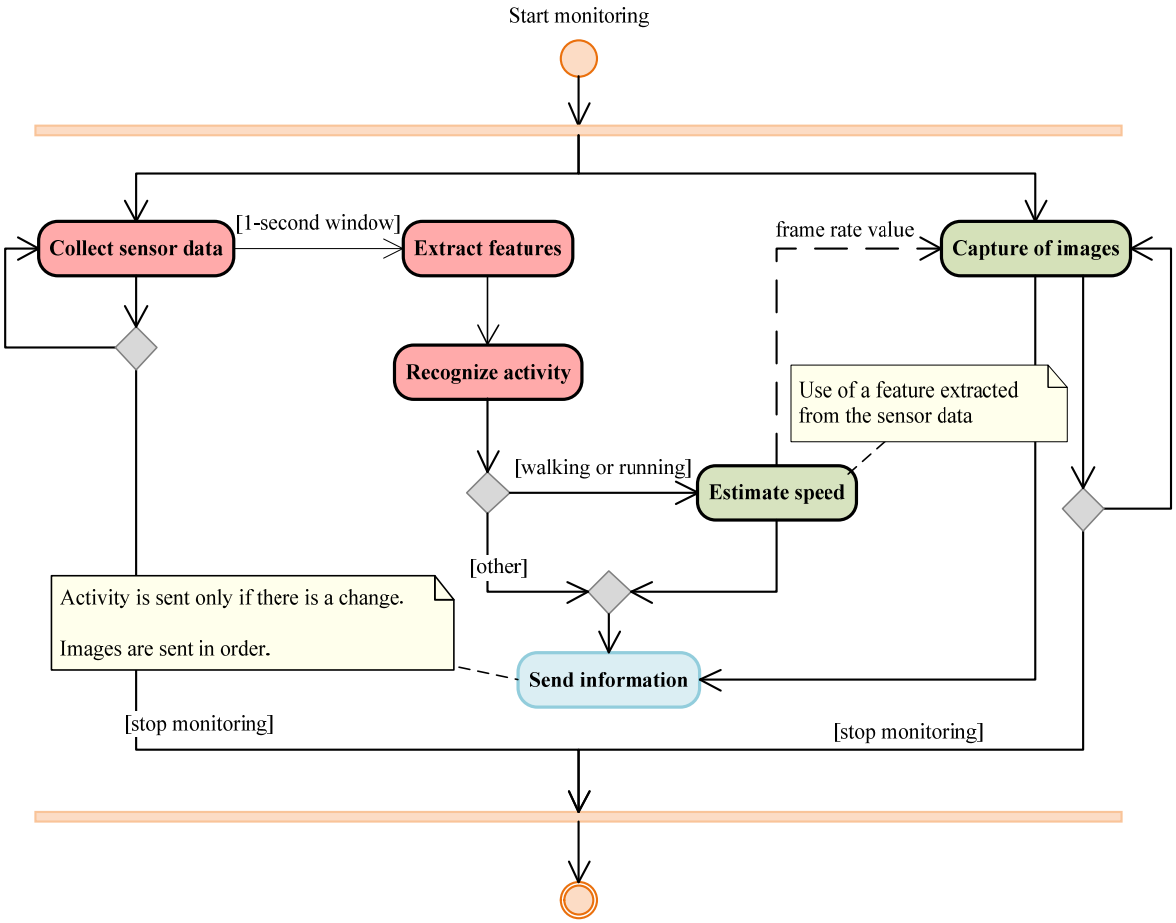


Figure 6. Activity diagram of the MonitorMe smartphone application.

Smartphone Constraints

The MonitorMe system is based on the use of a smartphone for achieving online monitoring. It was then necessary to take into account the restrictions associated with the use of mobile phones. Most of these devices have limited resources, namely low processing power when compared with desktop or portable computers. Even though some of the most recent smartphones models have quad-core 1.5 GHz processors, they still have a constrained battery life, which is greatly affected by the amount of processing performed. Therefore, it is important to achieve a balance between the accuracy of the developed algorithm for activity recognition and the required processing resources. Furthermore, for speed estimation the aim is to use as much as possible features calculated for activity recognition, rather than the phone's GPS (which represents extra resources and additionally can only be used in outdoor scenarios).

Another relevant aspect in the MonitorMe implementation is the system performance. Since its aim is online monitoring, it is important to minimize the delay of the information sent from the smartphone to the remote device. Besides the delays related with the processing in the smartphone, there are delays associated with the connections between the devices. The latter depends mainly of the type of connection being used, which can vary a lot. But a factor that can be controlled is the quantity of information sent, which was reduced by compressing the captured images. In addition,

the required transmission bandwidth was minimized by keeping the image rate at 1 frame per second (fps) if the user is carrying out a static activity. If the user is walking or running, the frame rate is set according to the estimated user speed. This also contributes to the minimization of the required smartphone processing.

3.2 MonitorMe Architecture

The setup of MonitorMe includes an Android-based smartphone (Samsung Galaxy S II) and a MARG module [5], both equipped with a tri-axial accelerometer and a tri-axial gyroscope. More details on their specifications can be found below. The smartphone is placed at the chest (in the left shirt pocket) with the primary camera pointing to the front (screen facing the user), and the module is placed at the leg (in the right pants pocket), as illustrated in Figure 7. For simplicity, in this dissertation the smartphone will sometimes be referred as "phone" and the MARG module as "module".

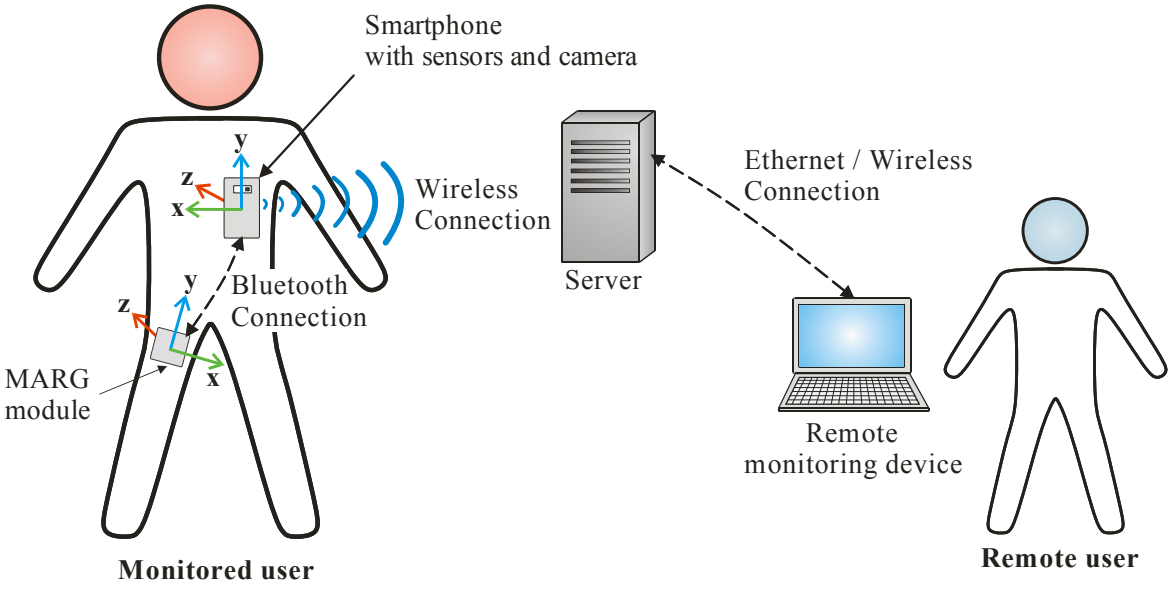


Figure 7. The MonitorMe system architecture, including the location of the smartphone and the MARG module device (and associated coordinate systems).

The module sends the data collected from its sensors to the phone through a Bluetooth link, initiated by the phone. The MonitorMe system also includes a server that receives the relevant information (i.e. the current recognized activity and the captured images) sent by the phone through a wireless data connection, such as Wi-Fi or 3G/4G (3^{rd}/4^{th} generation mobile telecommunications). If the remote monitoring device is connected to the server, an application running at this remote device will receive and present the referred information to its user, allowing online monitoring.

The main components of MonitorMe are the smartphone and the MARG module, since they are worn by the user to be monitored. Both include motion sensors that were used to develop the activity recognition and speed estimation algorithms. The smartphone also includes a camera that is

used for image capture. Some of the most relevant specifications of both devices are described below.

Smartphone

The model of the smartphone used in MonitorMe is the Samsung I9100 Galaxy S II. From the sensors available in this model, the tri-axial accelerometer and the tri-axial gyroscope were used in this dissertation for the MonitorMe development. The accelerometer is the LIS3DH model from STMicroelectronics [38], with a range of ±2*g* for the Galaxy S II. The gyroscope is the L3G4200D model also from STMicroelectronics [39], with a range of ±500 dps (degrees per second) for the Galaxy S II. The accelerometer and gyroscope data are measured in m/s^2 and rad/s, respectively. The coordinate system for each of the sensors is presented in Figure 8.

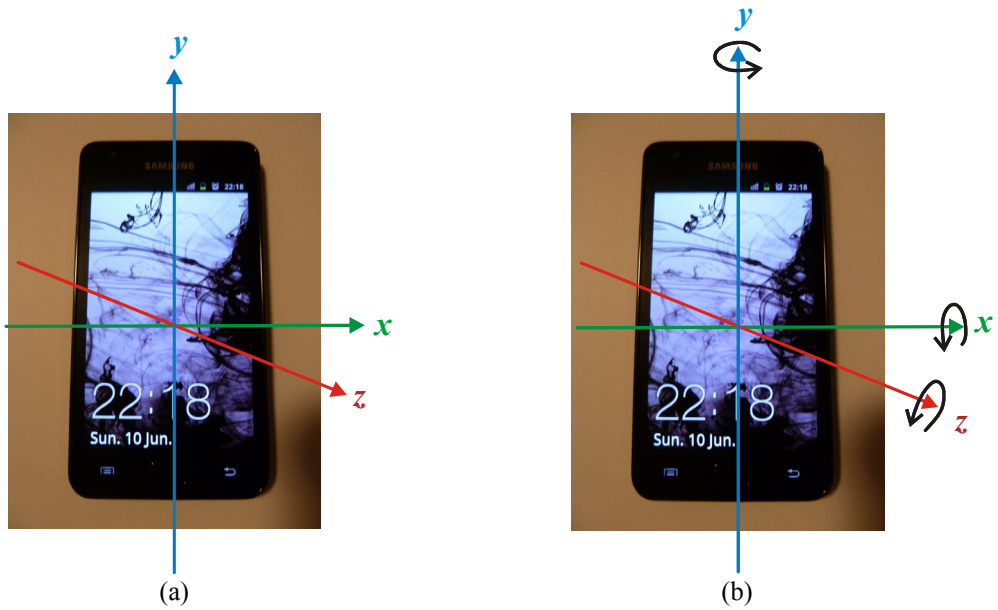


Figure 8. Axes orientation for the: (a) smartphone accelerometer; (b) smartphone gyroscope.

As explained previously in this dissertation, a tri-axial gyroscope measures the angular speed in rad/s (or dps) around a device's x-, y-, and z-axis. Regarding the smartphone, as shown in Figure 8 (b), the rotation is considered positive if an observer, looking from a given positive location on the x-, y- or z-axis at the device positioned at the origin, reports a counter-clockwise rotation.

Other Galaxy S II specifications [40, 41] that can be relevant for the operation of MonitorMe, more specifically for its Android application, are the presented in Table 3.

Table 3. Samsung Galaxy S II main specifications

Specification	Description
Processor	Dual core 1.2 GHz ARM Cortex A9
Memory (RAM)	1 gigabyte
Primary camera	8 megapixels, 3264×2448 pixels
Connectivity	– Wi-Fi 802.11 a/b/g/n – Bluetooth v3.0+HS
Battery	– Li-Ion 1650 mAh – Stand-by up to 710h (2G) or 610h (3G) – Talk time up to 18h20 (2G) or 8h40 (3G)
OS (Operating System)	Android 2.3.3

MARG Module

MonitorMe has the possibility of using another device that provides sensor data. In this dissertation an in-house built MARG module developed by the authors of [5, 42] was used. This module has a PIC24FJ64GA002 microcontroller from Microchip [43], a low power embedded Bluetooth v2.0+EDR (F2M03GLA from Free2Move [44]), and a tri-axial magnetometer, a tri-axial accelerometer and a tri-axial gyroscope. Similarly to the smartphone, the accelerometer and gyroscope of the MARG module were used in this dissertation for the development of MonitorMe. The accelerometer is the KXTF9-1026 model from Kionix [45], with a range of ±2*g*. The gyroscope is the ITG-3200 model from Invensense [46], with a range of ±2000 dps. The MARG module was programmed to send a string including sensor data, with an approximate rate of 25 Hz.

The coordinate systems associated with the MARG module sensors detailed above are presented in Figure 9. As it can be seen, the only differences in relation to the smartphone are the z-axis direction and the convention for the positive values of the angular speed around the y-axis. This speed is now considered positive if an observer, looking from a given positive location on the y-axis at the device positioned at the origin, reports a clockwise rotation. Apart from these differences, the values of the accelerometer and gyroscope in the MARG module have the same meaning as the corresponding values in the smartphone. Regarding the units, they are the same for the accelerometer data, but for the gyroscope data the units are dps (converted to rad/s before feature extraction in MonitorMe).

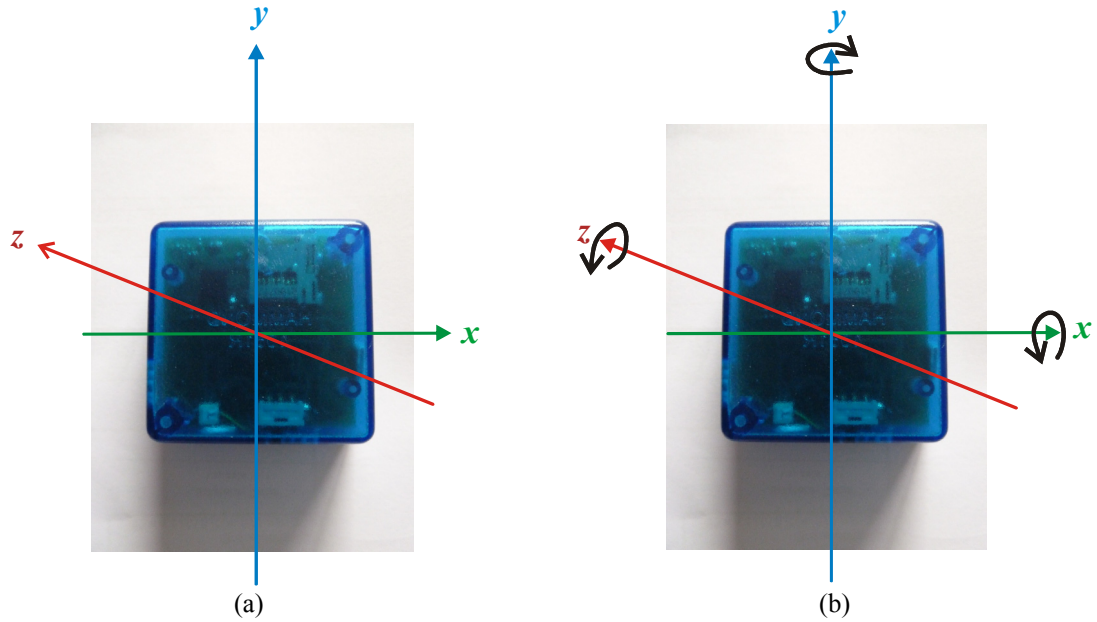


Figure 9. Axes orientation for the: (a) MARG module accelerometer; (b) MARG module gyroscope.

3.3 MonitorMe Implementation

3.3.1 Programming Environment

The operating system of the smartphone (Samsung Galaxy S II) used in this dissertation is the Android OS (Operating System). Android is an open-source, Linux-based OS for mobile devices (e.g. smartphones), developed by the Open Handset Alliance with the goal of allowing the easy development of new applications [47]. By March 2012, Android had the biggest share (48.5%) concerning U.S.A. smartphone owners [48].

As referred above (Table 3), the OS version of the Galaxy S II is the Android 2.3.3, which means that it can run applications developed for this or older versions. For developing an Android application it is necessary to use the Android SDK (Software Development Kit). From the available versions, the most adequate would then be the Android 2.3.3. But due to a bug related with the preview frames captured by the camera, the Android 2.2 was used. In addition, the Eclipse IDE (Integrated Development Environment) was used since it very easy to develop and test the Android applications using the ADT (Android Development Tools) Plugin for Eclipse.

Android provides APIs (Application Programming Interfaces) that enable the access to hardware features, such as the camera and other sensors, when supported by the Android-based device in use. This is very important for the implementation of the MonitorMe smartphone application. Using the Android sensor framework, it is possible to easily access the sensors integrated in the device and

collect sensor data, among other sensor-related tasks [7]. The Android framework also supports the capture of images though the Camera API.

The primary programing language used in the development of an Android application is a customized version of Java. This enables the reuse of the code that does not depend on the Android SDK, for other Java applications. For the MonitorMe applications that run on the server and on the remote monitoring device, Java was also the programming language used, together with the Eclipse IDE. This allows the reuse of the remote device application code (in Java), for the development of an Android application. This would mean that the remote monitoring device could be a desktop or a laptop computer (as illustrated in Figure **7**), but also a smartphone or a tablet for example.

3.3.2 Android Application

Most of the processing in the MonitorMe system is done by the Android application running on the smartphone. This application is organized in packages according to package diagram shown in Figure 10, where only the main packages are presented. In Figure 11 the most important classes of the application are presented, with the indication of the package they belong to (where "pt::ua" refers to the root package) and the associations between them. It is important to note that an association between class A and class B (excluding the relationship of inheritance) means that class A has a field of the data type B and calls a method of class B using that field. So, the associations between the presented classes (Figure 11) do not represent all the dependencies between the existing packages (Figure 10).

The "activities" package contains the main Android activity (corresponding to the MonitorMe class), which is associated with the user interface (UI) shown in Figure 12 (a). All main initializations take place in this activity (corresponding to the dependencies represented in Figure 10 by the light blue dashed lines originated in the "activities" package). This main activity also has an associated menu (shown at the bottom of Figure 12 (a)), which is presented to the user when the menu button of the smartphone is selected. The only option, "Connect Bluetooth device", is associated with another Android activity (BluetoothDeviceList class) that allows the management of Bluetooth connections through the UI shown in Figure 12 (b).

The "ui" package includes the classes that enable user interaction with the UI of the MonitorMe application (e.g. UI Handler class). As it can be seen from Figure 12, the UI allows the indication of the name of the user, the server IP address (for testing purposes) and whether or not the MARG module is to be used. It is also possible to start/stop monitoring by selecting the corresponding button. The start button initiates the sensor data gathering and processing ("ui" package dependencies with the "sensors" and "recognition" packages, shown in Figure 10) and the image capturing (dependency with the "camera" package). Also, if possible, a wireless connection is established with the server (dependency with the "communication" package). In Figure 10, all the mentioned dependencies are represented by the pink dashed lines originated in the "ui" package.

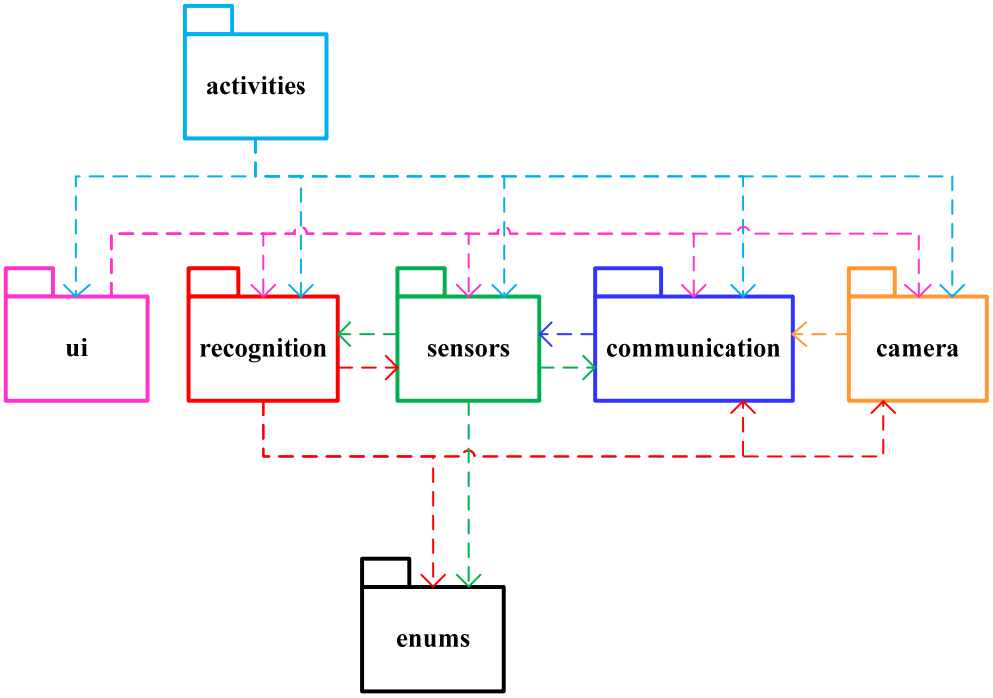


Figure 10. Package diagram of MonitorMe Android application.

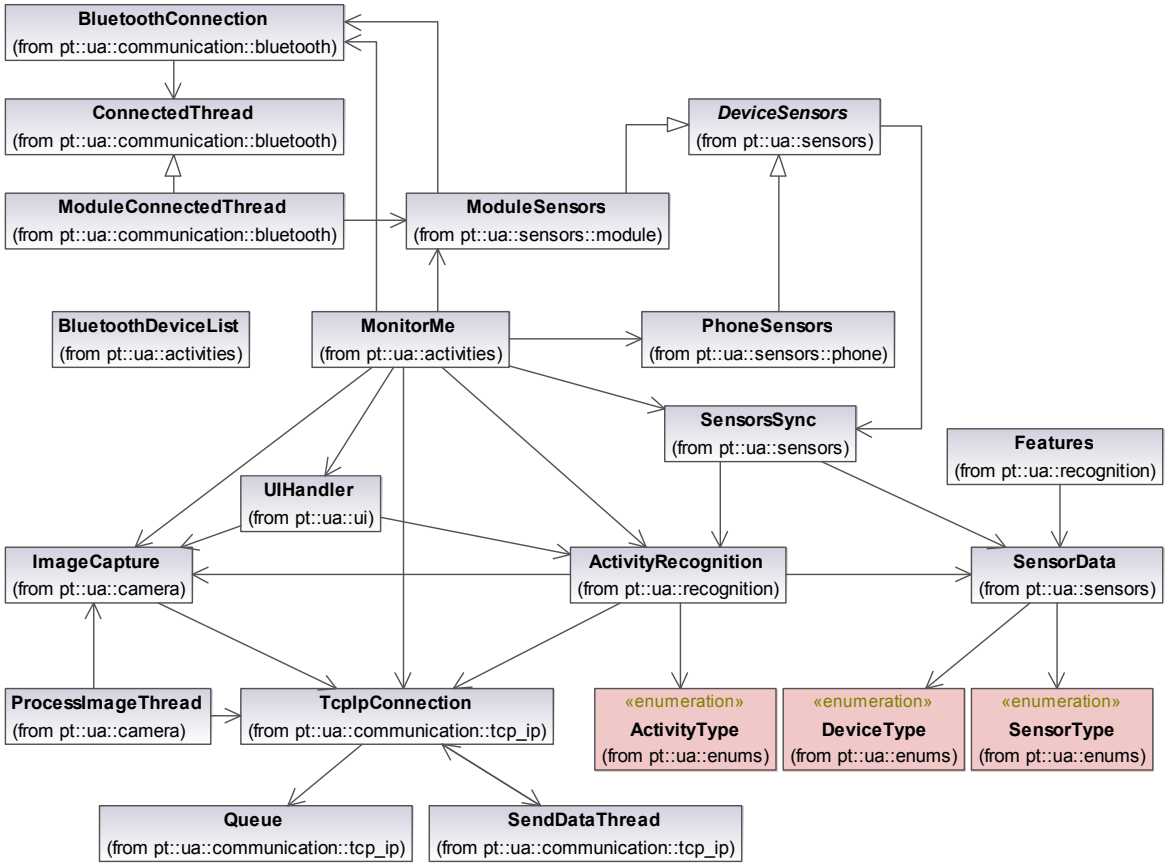
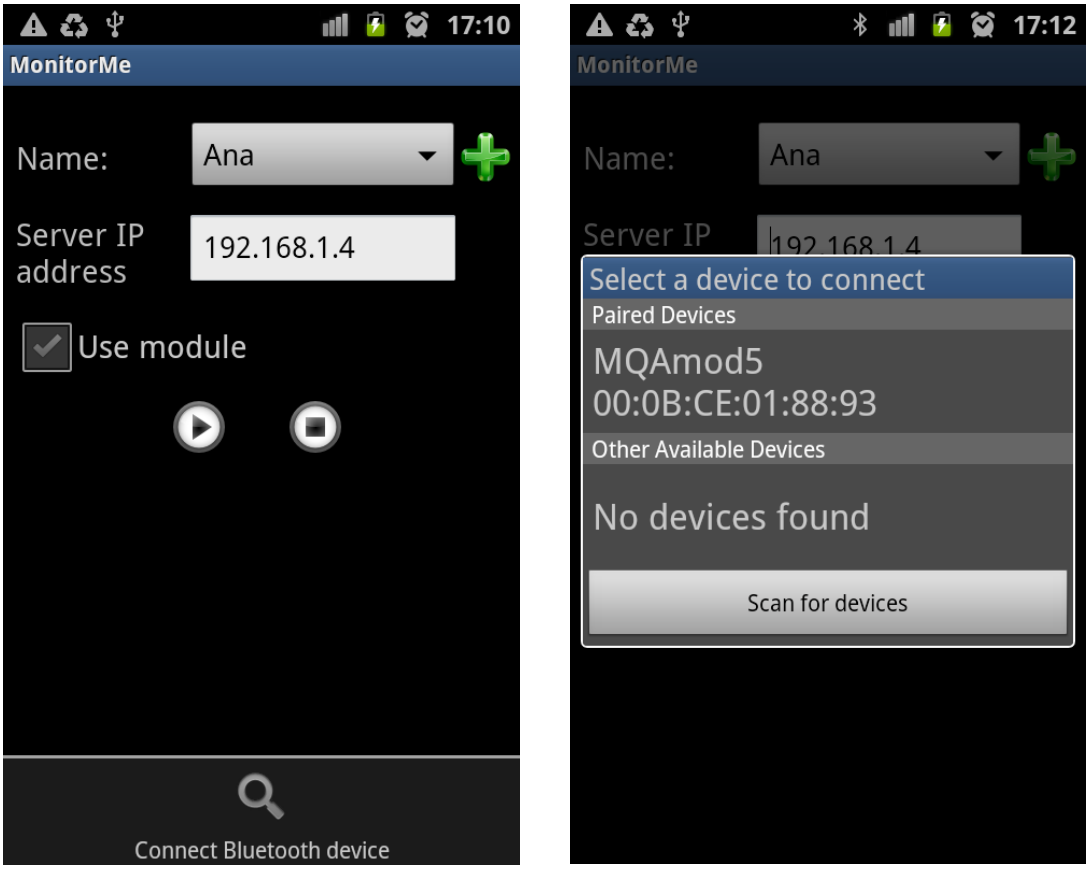


Figure 11. Class diagram of MonitorMe Android application.



(a) (b)

Figure 12. Capture of the smartphone screen when running the MonitorMe Android application.

The classes related with sensor data collection are included in the "sensors" package (e.g. DeviceSensors, SensorsSync and SensorData classes shown in Figure 11). MonitorMe gathers data from accelerometers integrated in an Android smartphone and/or in a device connected to the smartphone. For this reason, the "sensors" package has a dependency with the "enums" package, which includes the DeviceType enumeration and the SensorType enumeration (used by the SensorData class, as shown in Figure 11). If there is a connection established between the phone and the other device, the beginning/ending of data exchange between these devices is controlled in the "sensors" package but it also depends on the "communication" package (see BluetoothConnection class in Figure 11). Whenever data from all sensors and devices are collected, the associated values are passed on to the class ActivityRecognition of the "recognition" package, which performs the activity recognition for each 1-second sliding window. All dependencies of the "sensors" package with others packages are represented in Figure 10 by the green dashed lines originated in this package.

Regarding the "recognition" package, it includes the classes ActivityRecognition and Features that deal with sensor data. For each 1-second window, the necessary features for activity recognition are extracted. Then, according to the algorithm presented in Chapter 4, the current activity is identified (the possible activities are included in the ActivityType enumeration of "enums" package). If necessary, the value of the frame rate of the image capture is calculated and is set in the appropriate class of the "camera" package (association between ActivityRecongition and ImageCapture

classes, as seen shown in Figure 11). The activity information is sent to the server, using the class TcpIpConnection of the "communication" package that manages the associated connection. In Figure 10, the referred dependencies are represented by the red dashed lines originated in the "recognition" package.

The "camera" package consists in the classes that allow the capture of images using the smartphone camera (ImageCapture and ProcessImageThread). Similarly to the activity information, after processed, the data of each image is sent to the server (using the TcpIpConnection class). For this reason there is a dependency between the "camera" and "communication" packages, represented by the orange dashed line in Figure 10. The "communication" package includes two sub-packages: "bluetooth" and "tcp_ip". The dependency between the "communication" and "sensors" packages (represented by the dark blue dashed line in Figure 10) is due to the fact that after a Bluetooth connection is established, a thread listens to incoming data that it is passed on to a class of the "sensors" package (association between ModuleConnectedThread and ModuleSensors classes in Figure 11).

More details on the main tasks performed by the MonitorMe Android application (shown in Figure 6) are presented next.

Sensor Data Collection

When the option of start monitoring is selected, if the user chose to use the module and it was possible to successfully establish a Bluetooth connection with this device, data from both the phone and the module are collected. Otherwise, data are collected only from the phone. In both situations the sample rate is of approximately 25 Hz. This value was used to minimize the processing cost, while allowing a number of samples per second that does not compromise the accuracy of the activity recognition algorithm. Considering the related work described in the previous chapter, this seemed to be an adequate choice.

When the data is collected from different devices/sensors in parallel it is necessary to synchronize the data (implemented in SensorsSync class, shown in Figure 11), i.e. only when there is new value available from all device and sensor combinations they are actually collected (considering always the most recent values). The set of data of each collection is time stamped with the millisecond offset from the Epoch (January 1, 1970, 00:00:00.000 GMT). For each window of 1 second, with a 50% overlap between consecutive windows, the corresponding sensor data is provided as an input to the feature extraction task.

Feature Extraction, Activity Recognition and Speed Estimation

For each sliding window of approximately 1 second, the necessary features are extracted from the collected sensor data (in the Features class) and processed by the activity recognition algorithm (in the ActivityRecognition class). The development and evaluation of the activity recognition algorithm are described in detail in Chapter 4.

During online monitoring, if the current recognized activity is either walking or running, the user speed is estimated. The frame rate of the image capture is set according to the estimated speed value, which is calculated using a feature extracted for activity recognition. More details on this are provided in Section 4.3.2. If the current activity is not walking or running, then the frame rate is set to 1 fps, except in the case a fall where the frame rate value is maintained.

Image Capture

In parallel with the data collection and activity recognition, images are captured using the phone camera, with a frame rate that varies according to the user activity/speed (as explained above). Regarding the capture of images, in Android it is not possible to record a video while streaming it at the same time. On the other hand, it is possible to preview captured frames. Since it is not possible to directly control the frame rate, the method of dropping frames was used for this purpose. Each image frame is converted to the JPEG format, with 50% compression. JPEG is an image file format that has associated small/medium sizes, when compared with the TIFF format for example, but that still has a good/high quality [30].

The processing of each image frame is done in a separate thread (see ProcessImageThread class in Figure 11), but the frames are sent in order of capture to maintain a video-like stream, without any further frame dropping than the one performed for frame rate control. This is possible by attributing a value to each frame, named the frame number, which starts at 1 and is incremented by 1 every time a new image frame is captured.

Information Transmission

Whenever new activity or image information is available, it is added to a queue to be sent to the server (if there is an active wireless connection between this and the phone). This waiting queue (represented by the Queue class in Figure 11) is actually formed by two queues, one for the activities and another for the images. Each element of the activity queue has an associated timestamp and activity description. Each element of the images queue contains a timestamp and the encoded binary data. The frame number of the next frame to be sent is associated with the images queue. The activities are sorted by ascending order of the timestamp value, while the images are sorted by ascending order of the frame number.

A thread constantly checks if there is new information to be sent (see SendDataThread class in Figure 11). When there are elements available in both queues, the timestamps of the elements at the head of the queues are compared. The element with the lowest timestamp is removed from its queue and is sent to the server. The exception is when an image has the lowest timestamp, but the frame number does not match the frame number of the next frame to be sent. In this case, the next activity information is sent, if any is available (even if it has a higher timestamp).

The activity data is sent using the following format: “ACT” and a single line with the activity name, separated by a carriage return. The image data has a similar format, but the first line is “IMG_*size*”, where *size* is the number of characters of the second line, which has the image byte

information encoded in Base64. Base64 is a group of encoding schemes that represent binary data in an ASCII string format. In this case, encoding the image binary data ensures that it remains intact during transport.

If a remote monitoring device is connected to the server, it will receive this information. An application running at the remote device will extract the activity and image data, and present it to its user. For obtaining the image frame, it is necessary to further decode the Base64-encoded string. For this dissertation the remote device application was developed in Java, so it can be easily reused to develop an Android application.

4 MonitorMe Algorithm

MonitorMe aims at recognizing 6 different activities (standing, sitting, lying, walking, running and fall) and estimating speed for video control. To achieve these objectives, sensor data was collected from several users and then analyzed with the purpose of developing a threshold-based activity recognition algorithm and a speed estimation algorithm. The user speed is used to obtain a frame rate value for the video recorded during online monitoring. The development of both algorithms is described in the next sections.

4.1 Data and Methods

Two experimental procedures involving adult users were performed. For each of the two experiments (experiment 1 and 2), Table 4 includes the number of participating users, as well as their characterization in terms of the average and [minimum, maximum] values for age, weight, height and body mass index (BMI). Only one of the total 11 volunteers participated in both experiments. The activities performed in the experiments include the following: standing (A1), sitting (A2), lying (A3), walking (A4), running (A5) and fall (A6).

Table 4. Users' characterization in both experiments, including the average and [minimum, maximum] range values for age, weight, height and BMI

Experiment	Number of users	Sex (M/F)	Age	Weight (kg)	Height (m)	BMI (kg/m^2)
1	6	6/0	23 [22, 25]	74 [70, 85]	1.75 [1.65, 1.75]	24.4 [21.1, 27.4]
2	5	3/2	37 [23, 63]	70 [54, 78]	1.73 [1.65, 1.91]	23.6 [18.5, 27.6]

Experiment 1

The first experiment focused on the study of static activities or postures (standing, sitting and lying) and instantaneous activities (falls). The following sequence of activities was carried out: A1, A4, A1, A2, A4, A1, A2, A4, A1, A3, A4, A1, A6, A3 and A1. Every activity lasted for about 6 seconds, excluding walking (A4). The sitting activity (A2) was carried out twice, but using different types of chairs (with different heights). A sofa was used for the first lying activity (A3), while some cushions were used for the fall (A6). On the second lying activity the subject is lying on the floor (after the fall). Even though this experiment did not focus on dynamic activities, the last walking event in the sequence was used for the data analysis described below, since it had longer duration when compared with the other 3 walking events.

The sequence mentioned above was performed 5 times by each of the 6 volunteers that participated in this experiment. An example of the accelerometer and gyroscope data (from the phone and module) recorded during a sequence in this experiment is presented in Fig. 4 (b). The corresponding activity being carried out at each instant is indicated in Fig. 4 (a). The time elapsed, in seconds, since the beginning of the sequence is shown in both figures.

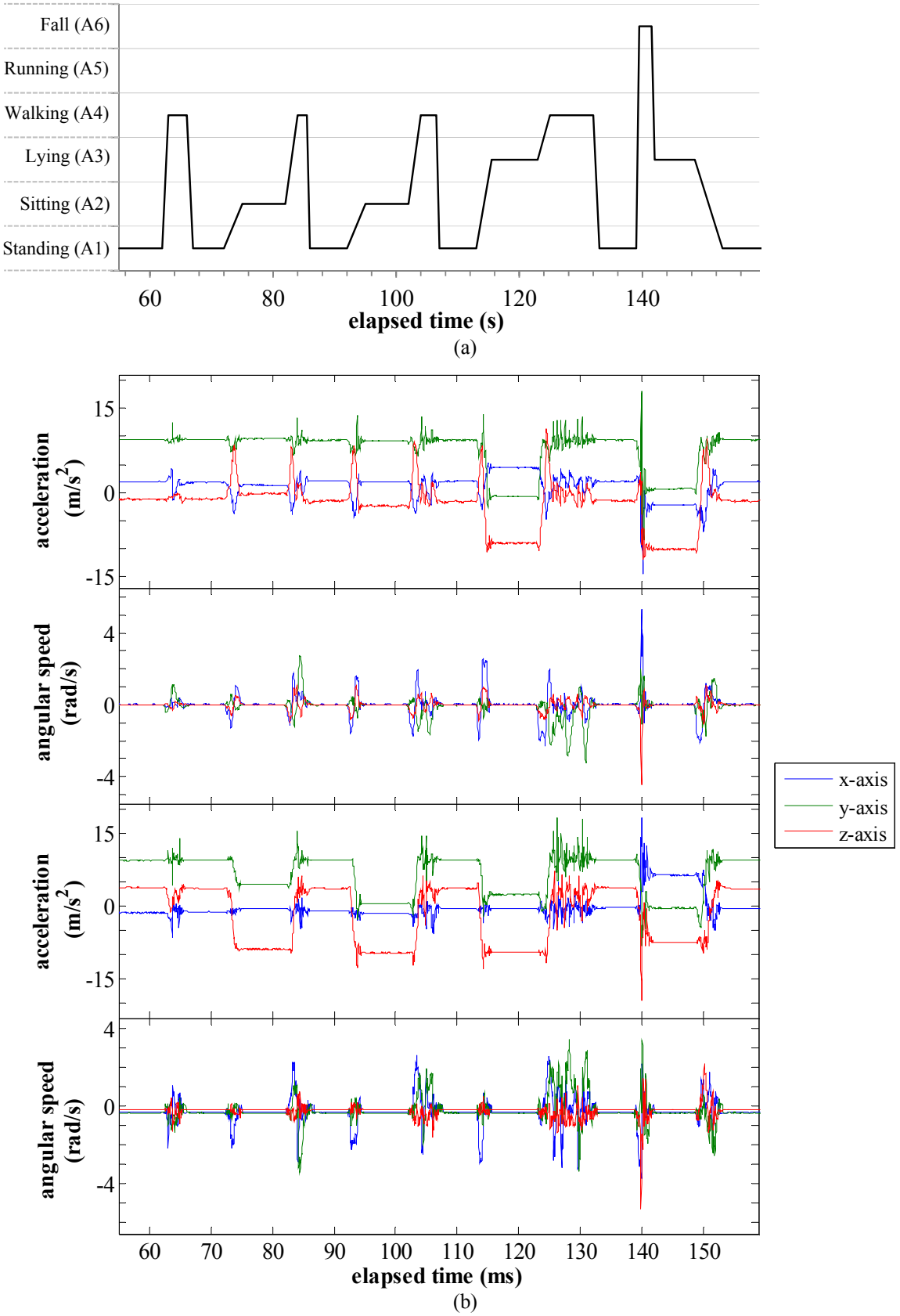


Figure 13. Example of a sequence performed in experiment 1: (a) activity being carried out at each instant; (b) data obtained from the phone accelerometer, phone gyroscope, module accelerometer and module gyroscope, respectively.

Experiment 2

The second experiment aimed at studying dynamic activities (walking and running) as well as the subject speed while performing them. Therefore, this experiment included a sequence of different ways of walking and running: walking slowly (W1), walking normally (W2), walking faster (W3), running slowly (R1) and running faster (R2). The sequence (W1, W2, W3, R1, R2) was repeated 4 times by each of the 5 volunteers that participated in this experiment. For each activity event, the duration of the covered distance of 18.5 m was recorded, so that an estimation of the average speed was possible.

Acquisition Setup

The acquisition setup used in the experiments included only the smartphone and the MARG module. Moreover, the phone application was different from the one described in Chapter 3, performing only the gathering of sensor data that were saved to files. The UI of this application (shown in Figure 14) allowed the indication of some extra information: the location of the devices (smartphone and MARG module), the activity or sequence of activities performed, and the smartphone sampling rate (the overall sampling rate depended not only on this value but also on the sampling rate of the module). This information was associated with the sensor data saved to a file.

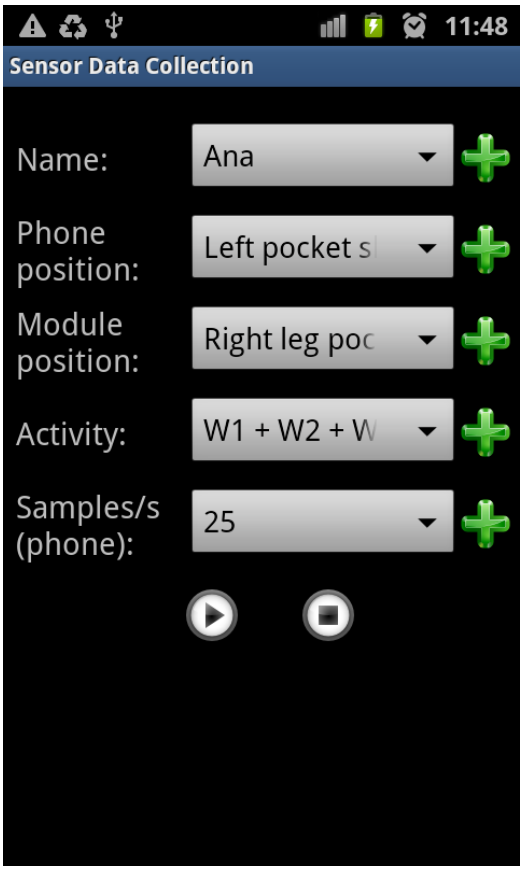


Figure 14. Capture of the smartphone screen when running the Android application for the sensors data collection.

Experimental Data Sets

For each sequence of consecutive activities performed in the experiments, data from the accelerometer and the gyroscope of both the smartphone and the MARG module were collected and saved to a file in the smartphone. The resulting set of files was transferred to a laptop and each file was split according to the different activities.

From the resulting files for each activity, 70% per subject were chosen randomly. Together these files formed a training set, used to develop a threshold-based activity recognition algorithm. The remaining 30% formed a test set used to evaluate the algorithm. For example, in the case of 10 files per user for the walking activity, 7 files out of total 10 would be chosen randomly from each user. For 5 users, this would lead to a training set of 7×5=35 files and a test set of (10–7)×5=15 files, for the walking activity.

Table 5 shows the number of events and the duration in seconds of the training and test sets as well as the total set, for each activity. For the development and evaluation of the speed estimation algorithm, only a fraction of these sets was used (walking and running data from experiment 2). The data analysis, detailed below for the activity recognition and the speed estimation algorithms, was carried out using Matlab and Microsoft Excel.

Table 5. Activity distribution in the training and test sets, represented by the number of events and the duration in seconds

		Standing (A1)	Sitting (A2)	Lying (A3)	Walking (A4)	Running (A5)	Fall (A6)	All
Training set	Number of events	115	40	39	62	28	19	303
	Duration (s)	588	233	224	630	152	33	1860 (≈ 31 min)
Test set	Number of events	53	18	18	27	12	9	137
	Duration (s)	288	102	106	279	67	12	854 (≈ 14 min)
Total set	Number of events	**168**	**58**	**57**	**89**	**40**	**28**	**440**
	Duration (s)	**876**	**335**	**330**	**909**	**219**	**45**	**2714 (≈ 45 min)**

4.2 Activity Recognition

To recognize activities based on sensor data, a threshold-based algorithm was developed. The next section (Section 4.2.1) presents the configuration of the devices and sensors, the features extracted from the sensor data and the associated thresholds, which are necessary to recognize the considered activities. The decision tree corresponding to the algorithm is also presented. A more detailed explanation of how the features and thresholds were chosen is included in Section 4.2.2.

4.2.1 Algorithm

An activity recognition algorithm was implemented based on thresholds and sliding windows of 1 second (25 samples), with an overlap of 50% between consecutive windows. In order to find the best configuration, different combinations of features, devices, sensors and axes were explored. For this purpose, it was necessary to perform the analysis of the sensor data collected in the experiments described above, which is presented in the next section. In the MonitorMe activity recognition algorithm, for each sliding window, the features listed in Table 6 are extracted and then the current activity is recognized according to the thresholds presented in Table 7. The algorithm is based on the decision tree represented by the flowchart shown in Figure 15.

Table 6 lists the features used in MonitorMe, including their description, the general expression used to compute them and the notation used to reference them in Table 7 and in Figure 15. Regarding the mean feature, its values are obtained for both devices (smartphone and module) and for each axis (x-, y-, and z- axis) separately. Table 7 summarizes the features and thresholds (values in terms of *g*) used to distinguish or identify the different activities, while Figure 15 shows how they are actually used.

In the equations presented in Table 6, x_i, y_i and z_i represent sample values of signals on the *x*-, *y*- and *z*-axis, respectively, and N is the number of samples per processing window. The terms $\min_j$ and mean_j, refer to the minimum and the mean value respectively, considering the *j*-axis, evaluated over each processing window.

Table 6. Features used in the activity recognition algorithm, including the corresponding expression and notation

Feature	Expression	Notation
Sum of minimum values on all axes of the phone accelerometer	$\min_x + \min_y + \min_z$	min_phone_acc
Sum of variances on all axes of the phone accelerometer	$\frac{\sum_{i=1}^{N}(x_i-\bar{x})^2}{N} + \frac{\sum_{i=1}^{N}(y_i-\bar{y})^2}{N} + \frac{\sum_{i=1}^{N}(z_i-\bar{z})^2}{N}$	var_phone_acc
Mean value in each of the axis of the phone & module accelerometer	$\text{mean}_x, \text{mean}_y, \text{mean}_z$	mean_*device*_acc_*axis* *

* *device* can be “phone” or “mod”, and *axis* can be x, y or z

Table 7. Features and thresholds used in the activity recognition algorithm

Activity	Feature	Thresholds
Fall	min_phone_acc	-2.294g
Static or dynamic	var_phone_acc	0.01g^2
Walking or running	mean_phone_acc_x, mean_phone_acc_y, mean_phone_acc_z	1.122g, 0.422g and -0.408g
	var_phone_acc	0.408g^2
Standing, sitting or lying	mean_phone_acc_y	1.122g, 0.62g and -0.51g
	mean_mod_acc_z	0.51g, -0.424g and -1.122g

The decision tree presented in Figure 15 shows how the activity corresponding to each 1-second window is recognized. As indicated in the flowchart, if some of the conditions are not met, the activity of the last window is used (if there is a previous window). It is important to note that this decision tree assumes the use of the module, to be able to distinguish standing from sitting. But if this last distinction is not necessary or important in a given monitoring environment, then it is possible to use just the smartphone. In this case, the algorithm is the same, with the exception of the conditions associated with the feature "mean_mod_acc_z" that do not exist. Moreover, if the condition "mean_phone_acc_y $\in$ [0.62g, 1.22g]" is true, then the output is "Standing or Sitting".

4.2.2 Data Analysis

To develop a threshold-based activity recognition algorithm, it was necessary to find the most adequate features and thresholds for identification of each of the considered activities. These can be divided in 3 different groups: static activities or postures (standing, sitting, lying), dynamic activities (walking, running) and instantaneous activities (fall). The analyzed features were extracted from the training set data, taking into account the entire duration of each activity event.

The choice of features to analyze, with the aim of identifying activities or distinguishing between groups of activities, took into account their main characteristics. During a fall there are higher/lower signal peaks when comparing with the other considered activities. Therefore, minimum and maximum values were explored. During static activities, the accelerometer sensor signals on each axis are normally stable around a given value, which can be different from posture to posture. So, the use of mean values seemed adequate for the purpose of distinguishing between static activities. To distinguish the latter from dynamic activities, features that translate the higher variation of signals during walking and running are more appropriate. This will be explained in more detail below.

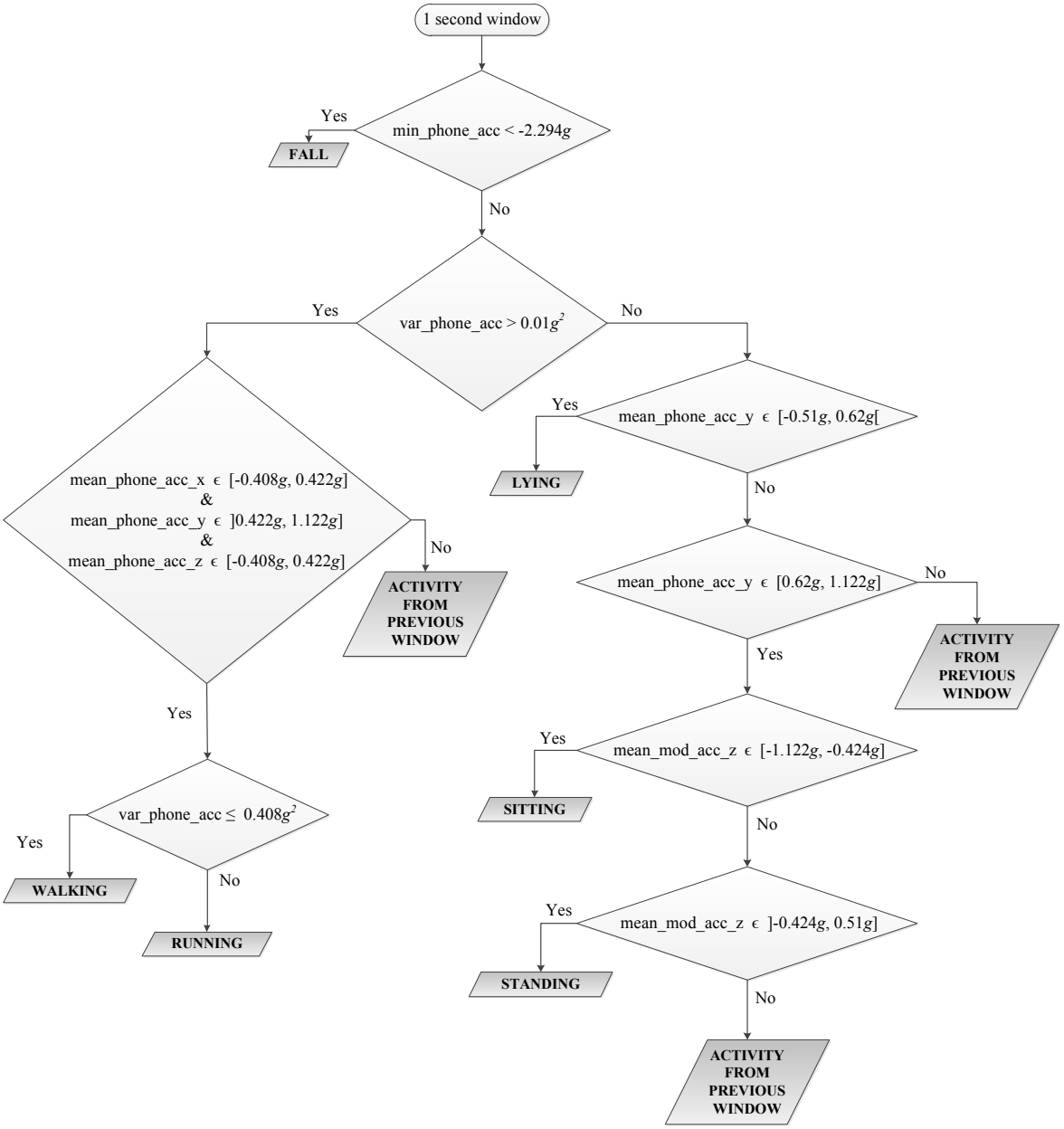


Figure 15. Decision tree that represents the activity recognition algorithm.

This section presents a summary of the analysis of features and the choice of thresholds that enable the recognition of the activities listed above. All features referred below were analyzed for both devices (smartphone and MARG module) and both sensors type (accelerometer and gyroscope). An exhaustive analysis was performed for various combinations of devices, sensors and features, but only the most relevant results are presented in this dissertation. Each choice took into account the possibility of using, whenever possible, only the phone (main device). Even though the acceleration data collected, analyzed and used in the developed algorithm was in m/s^2, the plots and thresholds associated with acceleration values are presented in this section in terms of *g*.

The range of values obtained for the various studied features are presented in the box plot format (e.g. Figure 16) implemented in Matlab [49]. In this format, the bottom and the top of a box represent the 25th and 75th percentiles of the data, respectively. The distance between the top and bottom values is denoted by interquartile range and the horizontal line inside a box is the median of the represented data (50th percentile). The vertical lines (whiskers) extending above and below the top and bottom of a box, respectively, indicate further values within the whisker length. Data beyond the whiskers lengths are marked as outliers. In the implementation used in this dissertation, an outlier (displayed with a red + sign) is a value that is more than 1.5 times the interquartile range away from the top or bottom of the box (default in Matlab).

Falls

A fall has the particular characteristic of being instantaneous, i.e. it normally happens in a much shorter interval of time than the other activities considered in this dissertation. Moreover, during this activity sensor signals usually have higher and/or lower peaks when comparing with the other activities. For this reason, the studied features for distinguishing between falls and other activities were the minimum value considering all axes (1) and the maximum value considering all axes (2). In the equations presented below, the terms $\min_i$ and $\max_i$, refer to the minimum and the maximum value respectively, considering the *i*-axis, where *i* can be *x*, *y* or *z*.

$$\min = \mathbf{minimum}\left(\min_x, \min_y, \min_z\right) \tag{1}$$

$$\max = \mathbf{maximum}\left(\max_x, \max_y, \max_z\right) \tag{2}$$

Figure 16 and Figure 17 show the boxplots associated with feature (1), representing the ranges of values obtained for each activity, for the phone accelerometer and module accelerometer respectively. It can be seen from Figure 16 (phone accelerometer) that there is no overlap between the values associated with falls and the values obtained for the other activities. However, Figure 17 shows that feature (1) obtained from the module accelerometer data is not suitable for fall detection, since there is overlap between the values for falls and the values for most of the other activities.

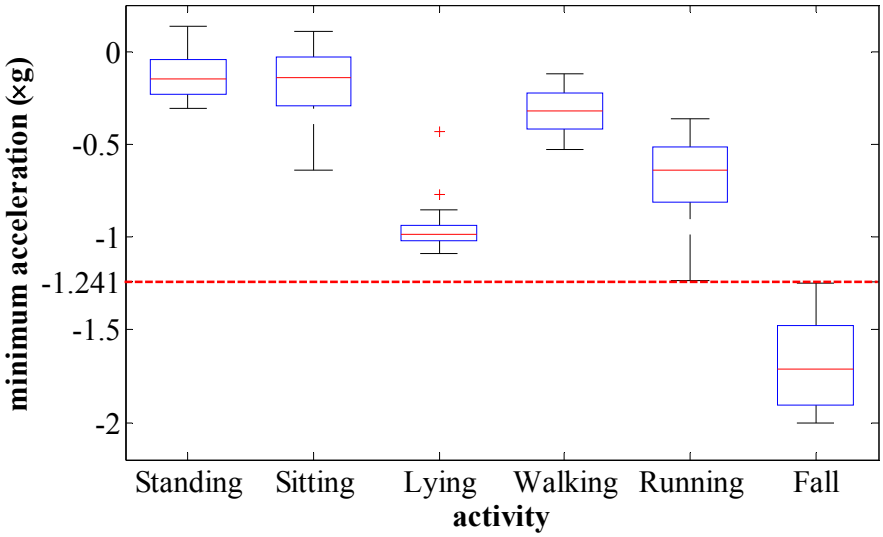


Figure 16. Box plot for the minimum value considering all three axes, according to (1), for the phone accelerometer.

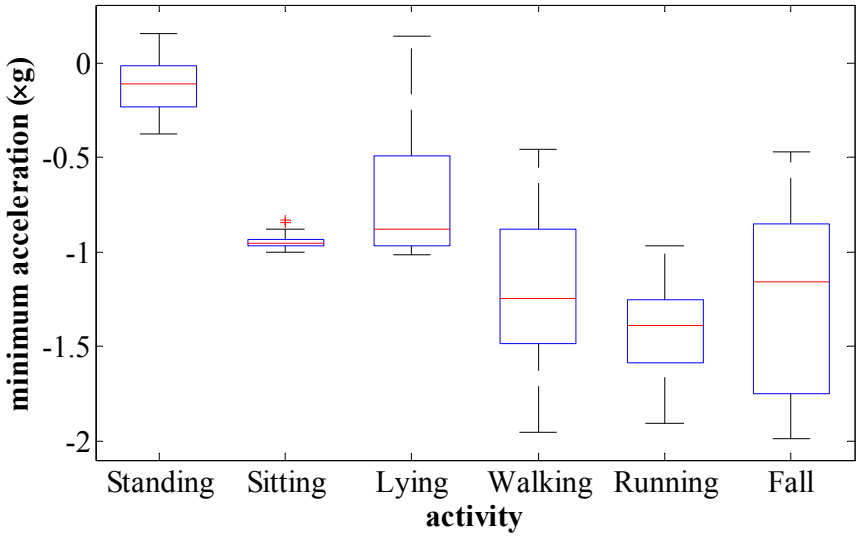


Figure 17. Box plot for the minimum value considering all three axes, according to (1), for the module accelerometer.

The boxplot of feature (2) for the phone accelerometer is presented in Figure 18. Figure 19 and Figure 20 show the boxplots for features (1) and (2), respectively, for the phone gyroscope. For all these 3 features, it can be seen that there is overlap between walking, running and fall values. So it is not possible to use them for identifying a fall. However, there is no overlap between the set of walking, running and fall activities and the postures (standing, sitting and lying), so they could possibly be used for distinguishing between static and dynamic activities (more details below).

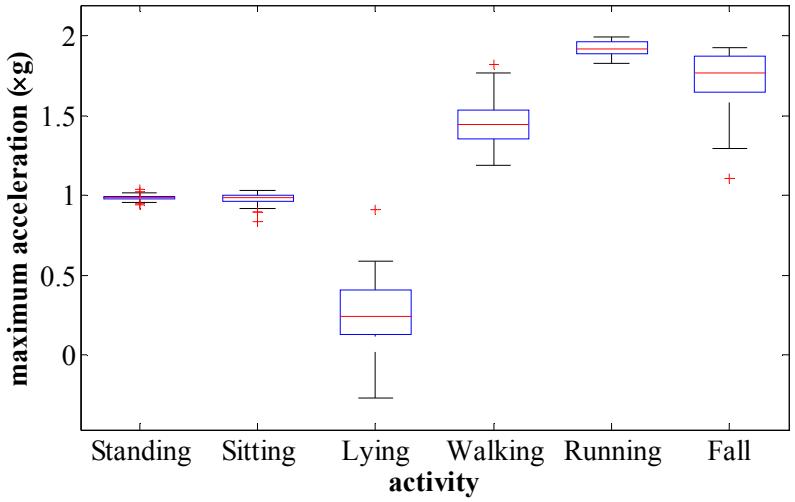


Figure 18. Box plot for the maximum value considering all three axes, according to (2), for the phone accelerometer.

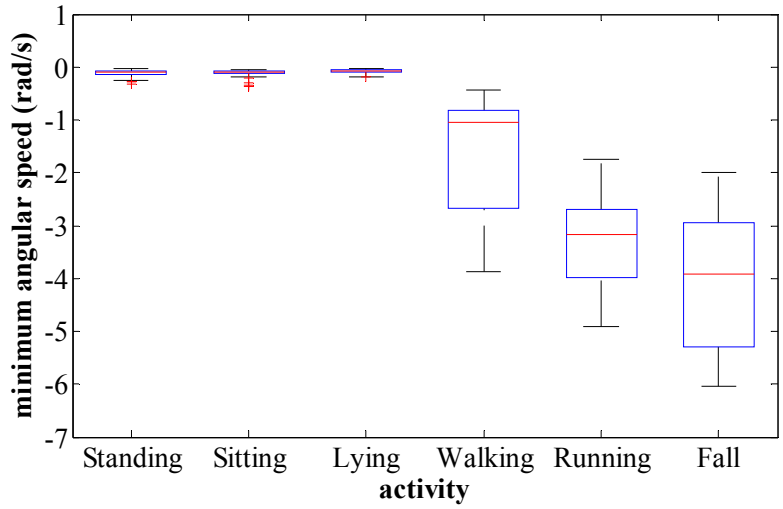


Figure 19. Box plot for the minimum value considering all three axes, according to (1), for the phone gyroscope.

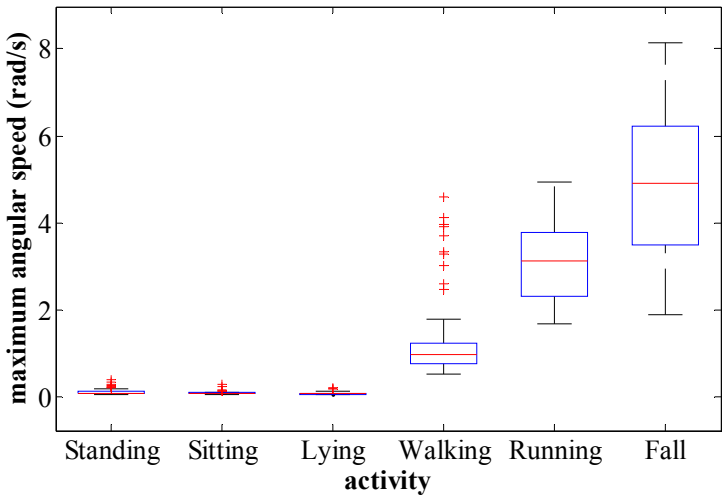


Figure 20. Box plot for the maximum value considering all three axes, according to (2), for the phone gyroscope.

Comparing all features considered above, it is possible to conclude that the most adequate for distinguishing falls from the other activities is the minimum value considering all axes (1) for the phone accelerometer. Associated with this feature, the threshold value of -1.241*g* was defined (Figure 16, red dashed line), corresponding to the midpoint between the maximum value for the falls and the minimum value for all other activities. A fall is then detected if the considered feature has a value lower than -1.241*g*.

Static versus Dynamic Activities

All activities, except the fall, belong either to the static or to the dynamic group of activities. The static activities correspond to postures where the subject is at rest or making minimal movements. On the other hand, dynamic activities are characterized by a larger amount of movement, which translates into higher variation of accelerometers and gyroscopes signals. The analysis concerning fall detection (above) has shown that some of the studied features may be used for distinguishing static from dynamic activities. However, they are not the most adequate, since the analysis considered the minimum or maximum value of an entire event, but the algorithm for activity recognition uses windows of just 1 second.

Some alternative features in which all samples contribute to their values, and that are intuitively interesting for the purpose of distinguishing static from dynamic activities, include the following: the sum of the mean squares on each axis (3), the sum of the variances on each axis (4) and the sum of mean absolute difference on each axis (5). In the equations presented below, x_i, y_i and z_i represent sample values of signals on the *x*-, *y*- and *z*-axis, respectively, $\bar{x}, \bar{y}, \bar{z}$ are the mean values of the x_i, y_i and z_i samples, respectively, and *N* is the number of samples considered in the calculations.

$$\text{sum mean sq.} = \frac{\sum_{i=1}^{N} x_i^2}{N} + \frac{\sum_{i=1}^{N} y_i^2}{N} + \frac{\sum_{i=1}^{N} z_i^2}{N} \tag{3}$$

$$\text{sum var.} = \frac{\sum_{i=1}^{N} \left(x_i - \bar{x}\right)^2}{N} + \frac{\sum_{i=1}^{N} \left(y_i - \bar{y}\right)^2}{N} + \frac{\sum_{i=1}^{N} \left(z_i - \bar{z}\right)^2}{N} \tag{4}$$

$$\text{sum mean abs. diff.} = \frac{\sum_{i=1}^{N} \left|x_i - \bar{x}\right|}{N} + \frac{\sum_{i=1}^{N} \left|y_i - \bar{y}\right|}{N} + \frac{\sum_{i=1}^{N} \left|z_i - \bar{z}\right|}{N} \tag{5}$$

These features can possibly help in distinguishing static from dynamic activities, since it is expected that during dynamic activities the signals (acceleration and angular speed) will have greater values (see signals in Figure 13 during walking activity). In feature (3) the squared values were used so that the signal variations are translated into meaningful feature values. If a simple mean was used, in some cases the negative values could cancel positive values and the variations in the signal might not be reflected into a corresponding variation of the feature value. The sum of the

mean square on all axes was considered, since the significant values can happen in any one of the three axes.

However, the boxplot corresponding to feature (3) for the phone accelerometer (Figure 21) shows that there is a clear overlap between the static activities and walking. This happens because the mean square takes into account the DC component of the signals. For instance, when in a standing position, the acceleration will have a strong component of approximately $+g$ on the axis perpendicular to the earth surface (Figure 13). The overlap of values between static and dynamic activities does not exist for the module accelerometer, since the amplitude of the legs motion is much bigger for dynamic than static activities, when comparing with the trunk motion (smartphone). The overlap between static and dynamic activities does not exist also for the phone gyroscope (Figure 22) and the module gyroscope. But in all these cases there is still an overlap between walking and running values.

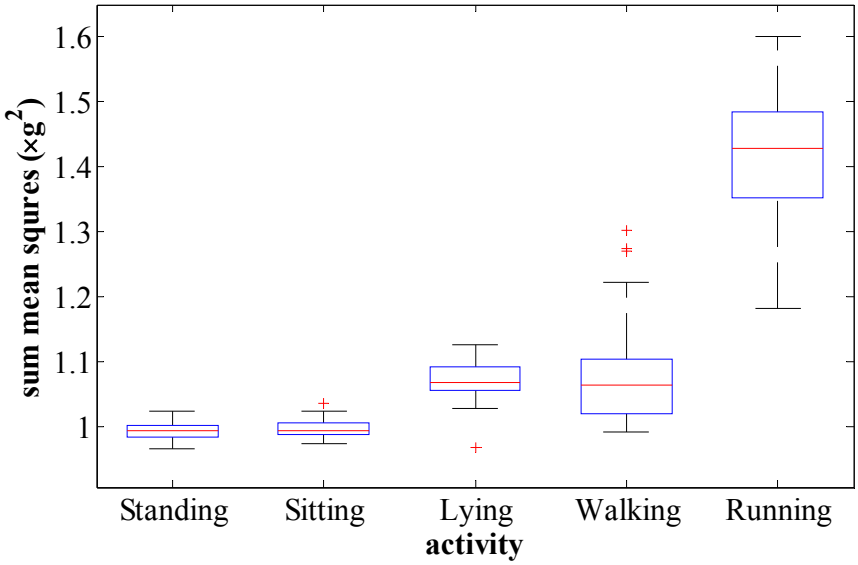


Figure 21. Box plot for the sum mean square values, according to (3), for the phone accelerometer.

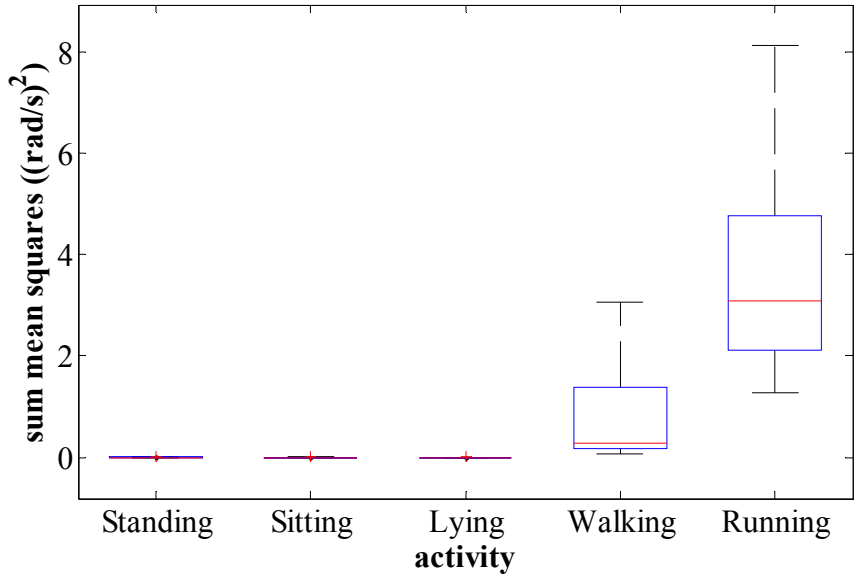


Figure 22. Box plot for the sum mean square values, according to (3), for the phone gyroscope.

Due to the difficulties originated by the DC component mentioned above, feature (4) was explored next. This feature takes into account just signal variations (independent of the DC component), since the difference between each value on an axis and the mean value on the same axis is computed. The results are better for this feature when comparing with (3), because there is no overlap between static and dynamic activities for all of the device and sensor combinations. This property is shown in Figure 23 for the phone accelerometer and in Figure 24 for the module accelerometer. Furthermore, for the phone accelerometer there is also no overlap between the waking and running activities.

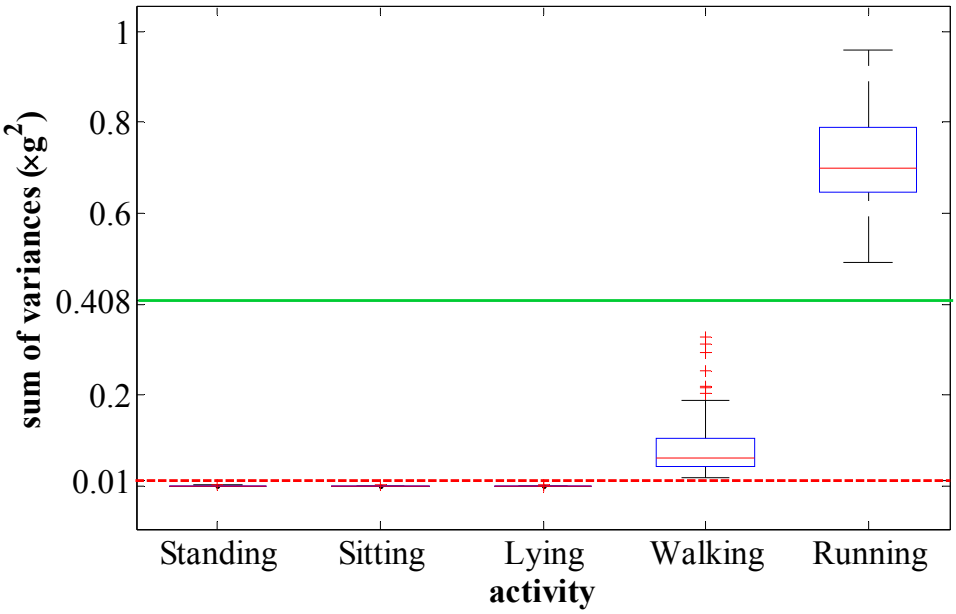


Figure 23. Box plot for the sum of variances values, according to (4), for the phone accelerometer.

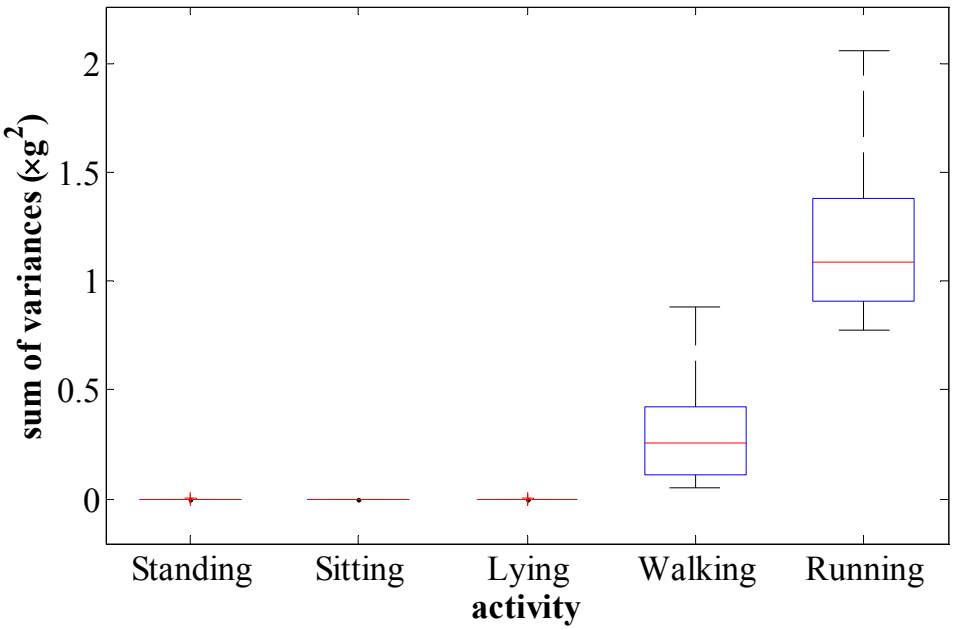


Figure 24. Box plot for the sum of variances values, according to (4), for the module accelerometer.

The variance on each axis alone was also analyzed. Some of the best results were obtained for the accelerometers y-axis, since when walking the acceleration on this axis will have positive value pulses, which correspond to the upward and forward forces which propel the body when walking. These forces are even greater for the running case. So the variance on the phone accelerometer y-axis, for example, could be used to distinguish static from dynamic activities. This feature has the additional advantage of less associated mathematical operations, when compared with the sum of the variances on each axis of the phone accelerometer. However, the latter feature depends less on the way each user walks or runs, so it is a better solution.

Feature (5) is similar to (4), but the absolute value of the difference between a value and the corresponding mean is calculated instead of the square of the same difference. The values obtained for feature (5) in the case of the phone accelerometer are represented by the boxplot in Figure 25. Comparing Figure 25 with Figure 23, it is possible to see that the ranges of values are different. However, the results concerning overlapping are similar for the two possibilities.

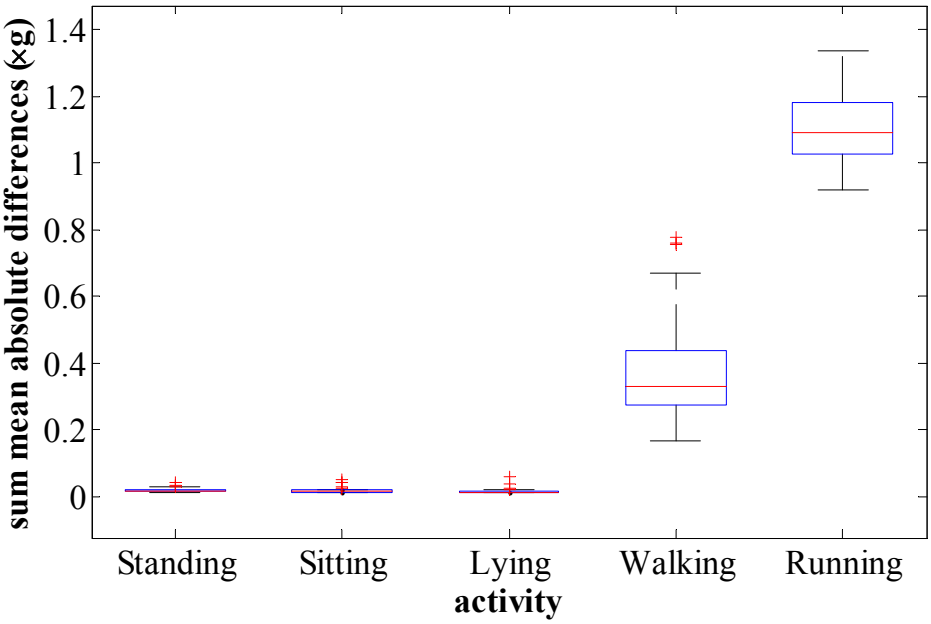


Figure 25. Box plot for the sum of mean absolute differences values, according to (5), for the phone accelerometer.

To quantify the quality of the two possibilities, the relative delta value defined in (5) was calculated for both features. The relative delta value for the sum of the variances on each axis of the phone accelerometer (value of 6.63) is considerably higher than the value of 1.85 obtained for the sum of absolute differences on each axis of the phone accelerometer. A similar result is observed when considering the ranges of values between the maximum value for walking and the minimum value for running. So despite the possible lower complexity of (5), the sum of variances feature for the phone accelerometer, was chosen for the purpose of distinguishing between static and dynamic activities.

$$\text{relative delta} = \frac{\mathbf{minimum}\left(\text{dynamic range values}\right) - \mathbf{maximum}\left(\text{static range values}\right)}{\mathbf{maximum}\left(\text{static range values}\right)} \tag{5}$$

So, to distinguish static from dynamic activities, a threshold associated with feature (4) was defined, which is represented in Figure 23 by the red dashed line. This threshold has a value of $0.01g^2$, corresponding to the midpoint between the maximum value for the static activities (standing, sitting and lying) and the minimum value for the dynamic activities (walking and running). If the value of the considered feature is equal or lower than $0.01g^2$ than the activity is static, else it is dynamic. In addition, this feature can also be used to distinguish between walking and running, as it will be further explained below.

Dynamic Activities

If an activity is identified as dynamic, it is necessary to confirm first if it can actually be either walking or running. This is done to help preventing, for example, the identification of transitions between activities as walking or running. This verification can be made by identifying the position of the trunk of the user, through the mean values on each axis of the phone accelerometer. Figure 26 presents the boxplot for these features. It is possible to see that none of the y-axis values is greater than $1.122g$ and that none of the x- and z-axis values is lower than $-0.408g$. Additionally, the value $0.422g$ divides the y-axis values from the x- and z-axis values, corresponding to the midpoint between the minimum value of the first group of values and the maximum values of the second group of values. So if the mean values on the x- and z-axis are in the range [$-0.408g$, $0.422g$] and the mean value on the y-axis is in the range]$0.422g$, $1.122g$], then the current activity identified previously as dynamic is confirmed as either walking or running. The defined thresholds of $1.122g$, $0.422g$ and $-0.408g$ are represented by red dashed lines in Figure 26.

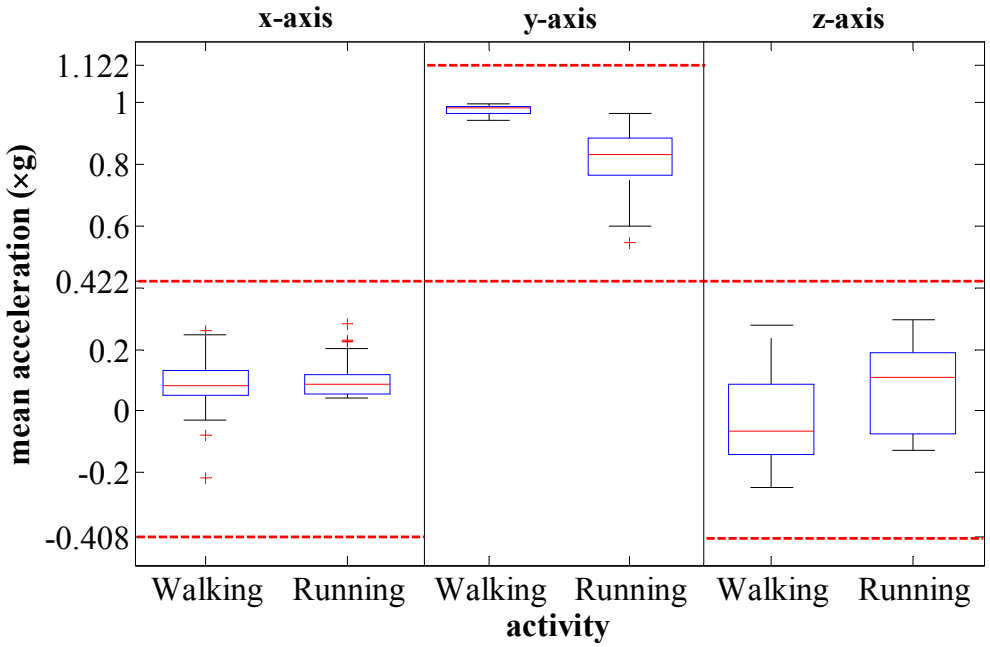


Figure 26. Box plot for the mean values on each axis of the phone accelerometer, for walking and running.

To actually distinguish walking from running, it is possible to use the sum of variances on each axis of the phone accelerometer, as mentioned above. The values obtained for this feature are shown in Figure 23. An activity is walking if the feature value is in the range $]0.01g^2, 0.408g^2]$ and is running if it is greater than $0.408g^2$. The defined threshold of $0.408g^2$ is represented in Figure 23 by a green solid line. This threshold corresponds to the midpoint between the maximum value for walking and the minimum value for running.

Static Activities

If an activity is identified as belonging to the group of static activities, it is necessary to verify if it is one of the considered postures. Since these are static activities, the obvious choice for a feature to distinguish between standing, sitting and lying is the mean value of acceleration on each axis. This choice can be explained by observing the signals in Figure 13 (b). As it can be seen, considering the phone accelerometer, the DC values on the x-, y- and z-axis for the standing/sitting postures are rather different from the equivalent values during a lying posture. On the other hand, in the case of the module accelerometer there is a great difference in the DC values for the y- and z-axis between the standing and sitting positions. A similar reasoning was used for verifying the trunk position during walking and running (above).

The boxplot that includes the values obtained for the mean values on each axis of the phone accelerometer is shown in Figure 27. From this figure it can be seen that the x- and z-axis values are useless for distinguishing between the static activities. However, it is possible to distinguish the lying posture from the other two postures by defining a threshold for the y-axis with a value of $0.62g$, which corresponds to the midpoint between the minimum value for standing and sitting and the maximum value for lying. Furthermore, considering the y-axis values, for both standing and sitting there is no value greater than $1.122g$, and for lying there is no value lower than $-0.51g$. So if the mean value on the phone accelerometer y-axis is in the range $[-0.51g, 0.62g[$, the activity is lying. Else if that value is in the range $[0.62g, 1.122g]$, the activity is either standing or sitting.

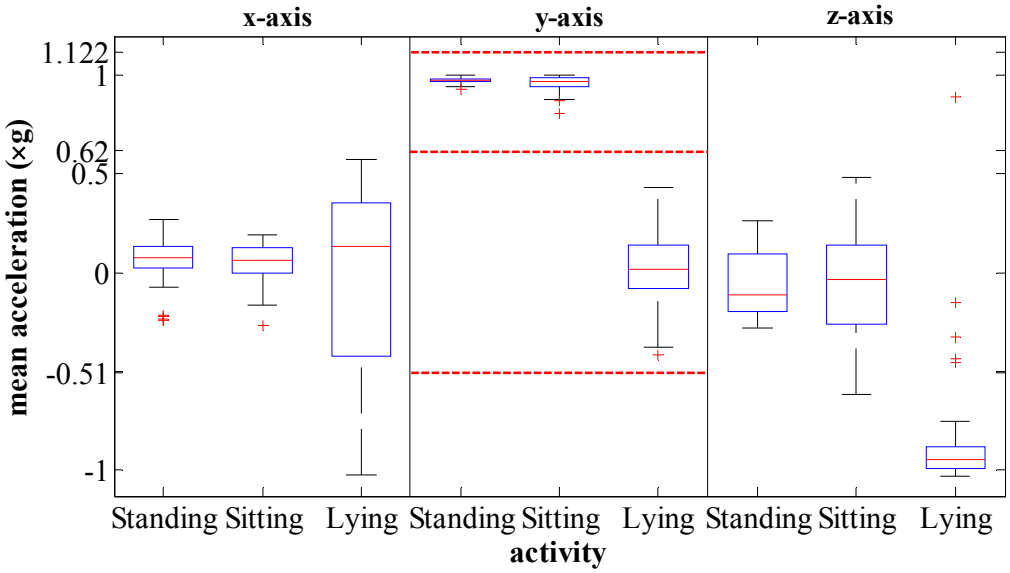


Figure 27. Mean values on the three axes of the phone accelerometer, for standing, sitting and lying postures.

To distinguish between standing and sitting it is necessary to use the module, whose associated mean values on each axis are represented by the boxplot shown in Figure 28. It can be seen that both the y- and z-axis values can be used to differentiate between these two postures, but the z-axis was the chosen one because it has a delta (6) of 7.32 m/s^2, while the same delta for the y-axis has a value of 4.507 m/s^2.

$$\text{delta} = \textbf{minimum}\left(\text{standing range values}\right) - \textbf{maximum}\left(\text{sitting range values}\right) \quad (6)$$

A threshold of -0.424*g* associated with the chosen feature was defined, corresponding to the midpoint between the minimum value for standing and maximum value for sitting. Furthermore, since there is no value greater than 0.51*g* and no value lower than -1.122*g* for the considered feature, the activity is considered as standing if the mean value of the module accelerometer on the z-axis is in the range]-0.424*g*, 0.51*g*] and as sitting if it is in the range [-1.122*g*, -0.424*g*]. All the thresholds used for identifying the different postures are represented by the red dashed lines in Figure 27 and Figure 28.

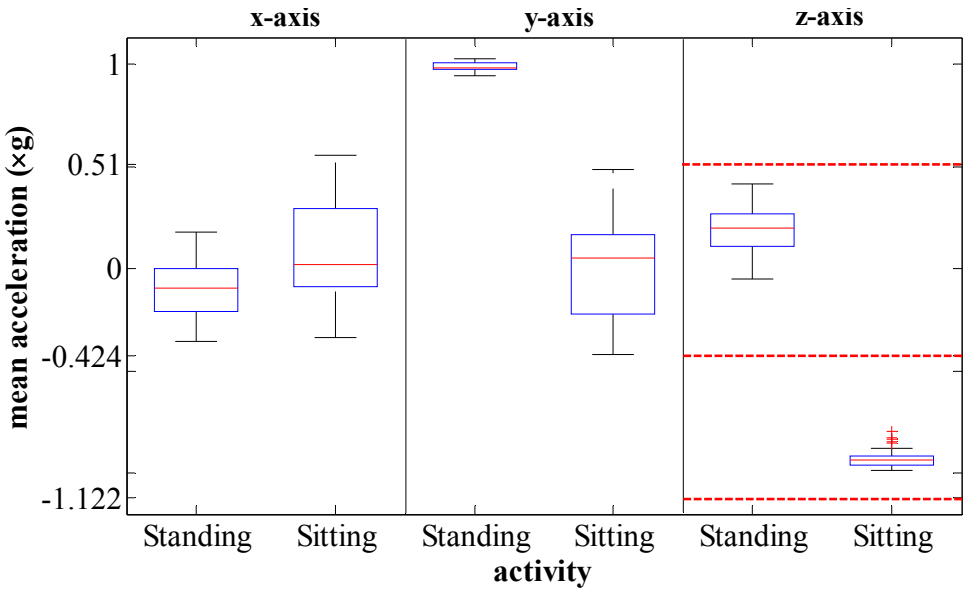


Figure 28. Mean values on the three axes of the module accelerometer, for standing and sitting postures.

Transitions

Besides recognizing activities, it is also necessary to deal with transitions between these activities. However, the aim of MonitorMe is not transitions detection (except falls). So, one way of dealing with these events is to maintain the current decision until another of the considered activities is actually recognized. This is achieved by adding the following condition the algorithm presented above: the current recognized activity is changed only when three consecutive decisions are the same.

Figure 29 shows an example where the 1-second sliding windows are represented by the "blocks" above the horizontal line ending with an arrow that corresponds to the time elapsed. Since overlapping windows are used, consecutive windows are represented alternately by blocks in the

first and second rows. Inside each block the actual activity being performed is indicated, unless there was a transition between activities (in which case the word "transition" is used). If the algorithm did not take into account the possibility of transitions, the recognized activities would be for example the ones shown below the vertical dashed lines. Using the method to deal with transitions mentioned above, the output of the activity recognition algorithm for the presented example corresponds to the activities shown below the vertical solid lines ending in an arrow.

As it can be seen from Figure 29, the decisions during the transition are ignored and the lying activity is recognized only after three consecutive decisions correspond to this activity. The final result is then correct with only a small delay of 1.5 seconds in detecting the lying activity, which is not important in many applications.

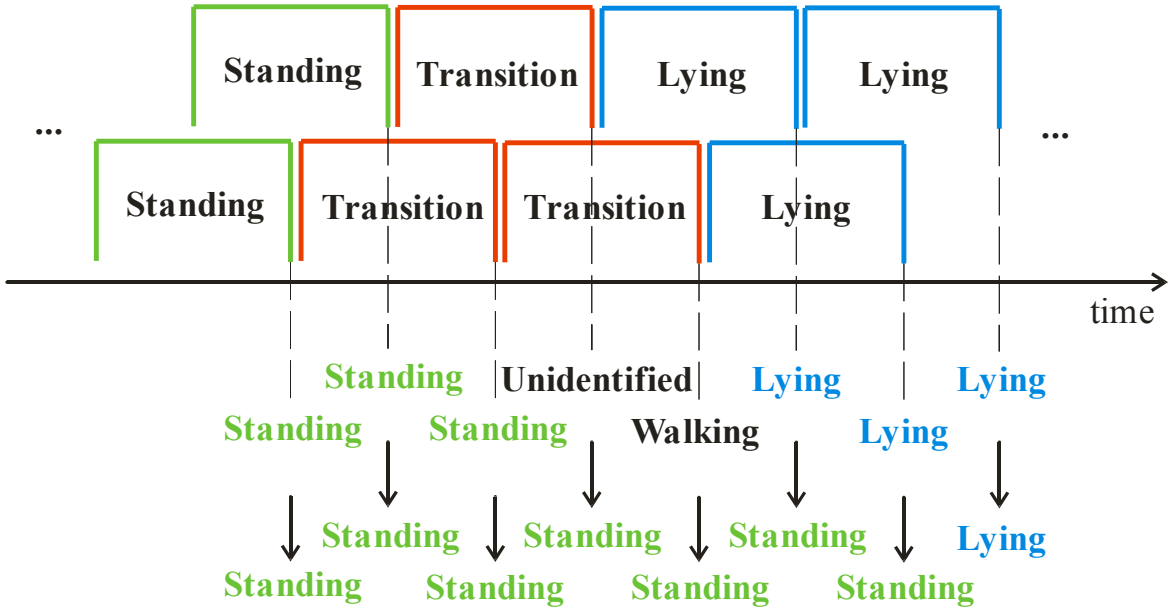


Figure 29. Example of how the transitions are dealt with in MonitorMe.

A study, using the original sequences collected during the first experiment, showed that most of the times this method produces the desired results. However, some transitions can be erroneously recognized as one of the considered activities for three or more consecutive windows, leading to a false positive. For the sequences of activities obtained in the first experiment, this situation occurs mainly in the transitions between the following postures:

- Standing to sitting (confused with walking)
- Standing to lying (confused with walking and fall)
- Lying (on the floor) to standing (confused with walking and fall)

Observing the accelerometer signals it is possible to see that some transition signals are very similar to the ones of walking or fall activities. Consequently, in these situations transitions sometimes have features values similar to the ones used to recognize walking or falls. Regarding transitions that are confused with walking, the number of mean crossings was analyzed (using 1-second windows, instead of considering the whole event). However, it was possible conclude that

this feature is not suitable for solving this problem. A further analysis of sensor data is needed to address these transitions.

Concerning falls, an analysis using 1-second windows was made. Some features were explored, including the sum of minimum values on each axis (6) and the sum of maximum values on each axis (7). Other features analyzed were the sum of the absolute values of the maximum and minimum values on each axis (8), and the sum of the absolute values of the minimum value and the maximum value considering all axes (9).

$$\text{sum minimums} = \mathbf{min}_x + \mathbf{min}_y + \mathbf{min}_z \tag{6}$$

$$\text{sum maximums} = \mathbf{max}_x + \mathbf{max}_y + \mathbf{max}_z \tag{7}$$

$$\text{sum maximums minimums} = \left|\mathbf{max}_x\right| + \left|\mathbf{max}_y\right| + \left|\mathbf{max}_z\right| + \left|\mathbf{min}_x\right| + \left|\mathbf{min}_y\right| + \left|\mathbf{min}_z\right| \tag{8}$$

$$\text{sum maximum minimum} = \left|\mathbf{max}\left(\mathbf{max}_x, \mathbf{max}_y, \mathbf{max}_z\right)\right| + \left|\mathbf{min}\left(\mathbf{min}_x, \mathbf{min}_y, \mathbf{min}_z\right)\right| \tag{9}$$

For each fall event of the training set, the minimum value for feature (6) and the maximum value for features (7), (8) and (9) were found. Considering the transitions where a fall is incorrectly detected (from the original sequences of experiment 1), the best results are obtained for features (6) and (8). Figure 30 and Figure 31 present the boxplots showing the range of values obtained for these features in the case of fall events and of transitions confused with falls. These figures show that it is not possible to eliminate completely the problem of false fall positives associated with some transitions. The difficulty of simulating real falls in the experiments may have also contributed to this problem. Nevertheless, these false fall positives can be minimized by defining a proper threshold.

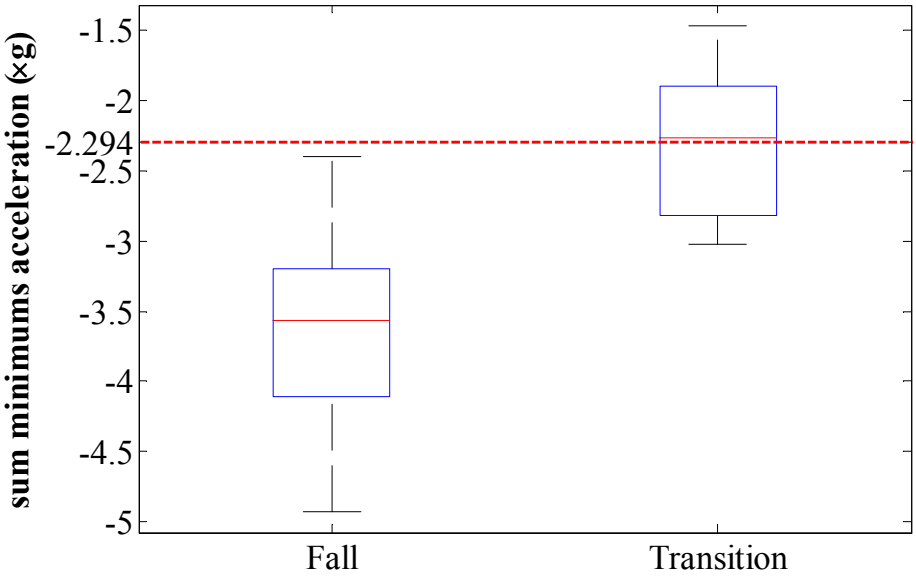


Figure 30. Range of values obtained for the sum of minimums on each axes of the phone accelerometer, according to (6), considering falls and transitions confused with falls.

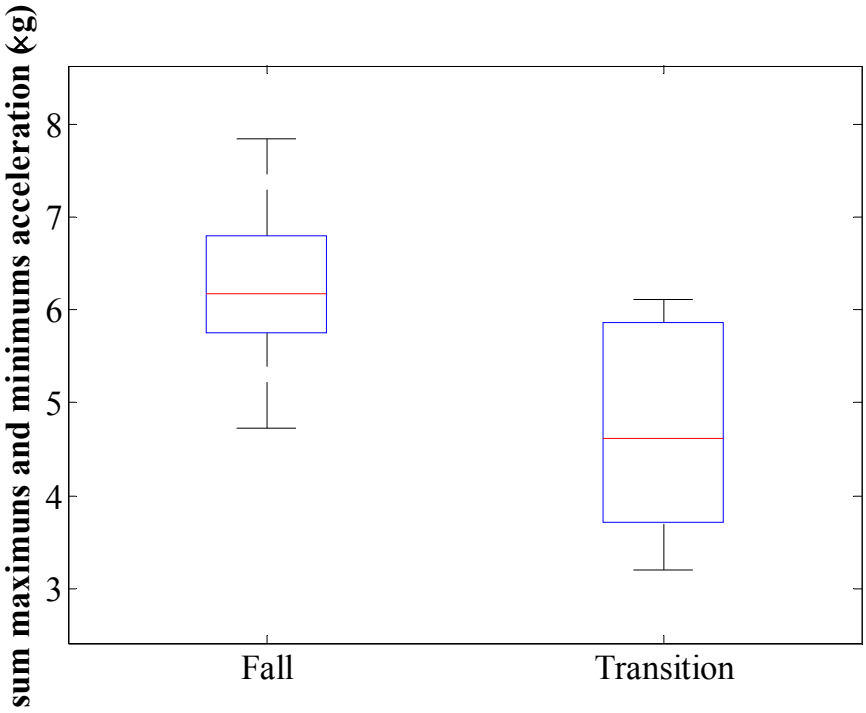


Figure 31. Range of values obtained for the sum of maximums and minimums on each axis axes of the phone accelerometer, according to (8), considering falls and transitions confused with falls.

For both features (6) and (8), it is possible to solve the transition problem (avoiding false fall positives) for 4 out of 7 of the considered transitions. However, the chosen feature was (6), since it does not need the additional computation of the maximums. The threshold defined to detect falls was then changed from the value of -1.241*g* associated with the minimum value considering all axes, defined previously without considering the transition problem, to a value of -2.294*g* associated with feature (6). This new threshold is represented in Figure 30 by a red dashed line. The threshold value was chosen so that all falls are detected and the greatest possible number of transitions is not confused with falls (it is better to have some false positives, rather than non-detected falls). Furthermore, none of the other activities (standing, sitting, lying, walking and running) are confused with a fall, since the minimum value of (6) obtained for this set of activities, considering the training set, is of -1.794*g*, a value well above the -2.294*g* threshold. So now a fall is detected if the sum of minimum values on each axis has value lower than -2.294*g*.

4.3 Speed Estimation

One of the objectives of this dissertation is to enable the rate control of the images captured by the phone camera during online monitoring, with the aim of saving phone resources and also reducing the required data transmission bandwidth. The next section explains how the video frame rate can be controlled using an estimation of the user speed. This is followed by the data analysis performed in order to obtain a speed estimator.

4.3.1 Video Control

In MonitorMe, the smartphone camera is used for video capturing. Furthermore, the frame rate of the video being recorded is varied according to the current activity/speed. If the current recognized activity is walking or running, the frame rate is set according to the user speed. Otherwise, the frame rate is set to 1 fps, except during a fall where the frame rate value is maintained. This enables the minimization of required transmission bandwidth when the user is not very active and also the adaptation of the frame rate value to the user speed when he/she is moving around.

So when the user is walking or running, the video frame rate can be defined as a function of speed using (9). The parameters A and B can be set in accordance with the requirements of the specific application environment. For the MonitorMe system presented in this dissertation, the values of A=1 m^{-1} and B=1 s^{-1} were considered. More specifically, the value for the parameter B was chosen so that the frame rate value for dynamic activities is always greater than its value for the other activities. The video frame rate is set to the calculated frame rate value rounded to the unit.

$$\mathbf{frame\_rate}\left(speed\right) = A \times speed + B \tag{9}$$

4.3.2 Data Analysis

The data analyzed for speed estimation was obtained in the second experiment, which included walking and running events with different associated average speeds. For each device-sensor combination, several features were explored, most of them already analyzed for distinguishing static from dynamic activities and walking from running. Also here the analysis took into account the entire duration of each activity event.

For each feature, a scatter plot of the average speed versus the feature value was obtained and then a linear regression was performed. The choice of this type of regression is due to the smaller number of operations necessary (only a multiplication and a sum) when comparing to exponential or polynomial regressions of higher order. The coefficient of determination (R^2) associated with each linear regression is presented in the corresponding scatter plot. The acceleration values presented in this section are in m/s^2 (they are not expressed in terms of g, as in the section 4.2).

For the sum of mean squares on each axis (3), the scatter plots obtained for the phone and module accelerometer are shown in Figure 32 and Figure 33, respectively. For both plots there are some points corresponding to the highest speeds that have a significantly lower value for the feature when compared with some lower speed cases. An example of this reversal is given in Figure 32, where point A refers to an event where a subject is running with a higher speed than the speed of the subject in event B. However, as can be seen in the figure, the value of the sum of mean squares for event A is significantly lower than this feature value for event B. So the mean square feature is not appropriate for speed estimation.

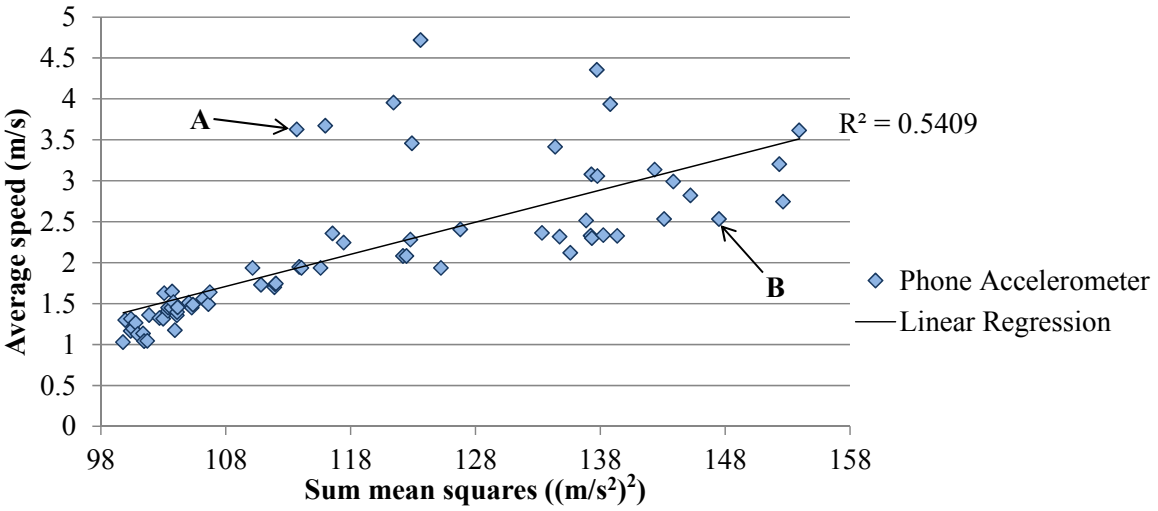


Figure 32. Scatter plot of the average speed versus the sum of mean squares on each axis of the phone accelerometer, including the corresponding linear regression. Points A and B refer to two different events. In event A a subject was running with higher speed than the speed corresponding to the subject in event B.

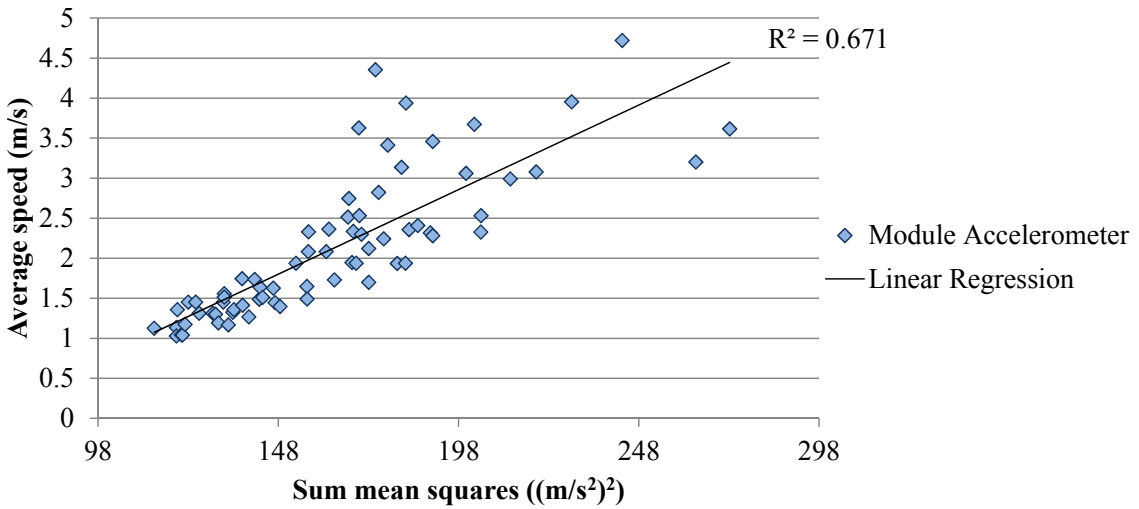


Figure 33. Scatter plot of the average speed versus the sum of mean squares on each axis of the module accelerometer, including the corresponding linear regression.

The sum of mean absolute differences on each axis (5) was also studied. The corresponding results are shown in the scatter plots of Figure 34 (phone accelerometer) and Figure 35 (module accelerometer). The coefficients of determination associated with the linear regressions have higher values when compared with the sum of mean squares on each axis (0.8682 and 0.8541, instead of 0.5409 and 0.671, for the phone and module accelerometers respectively).

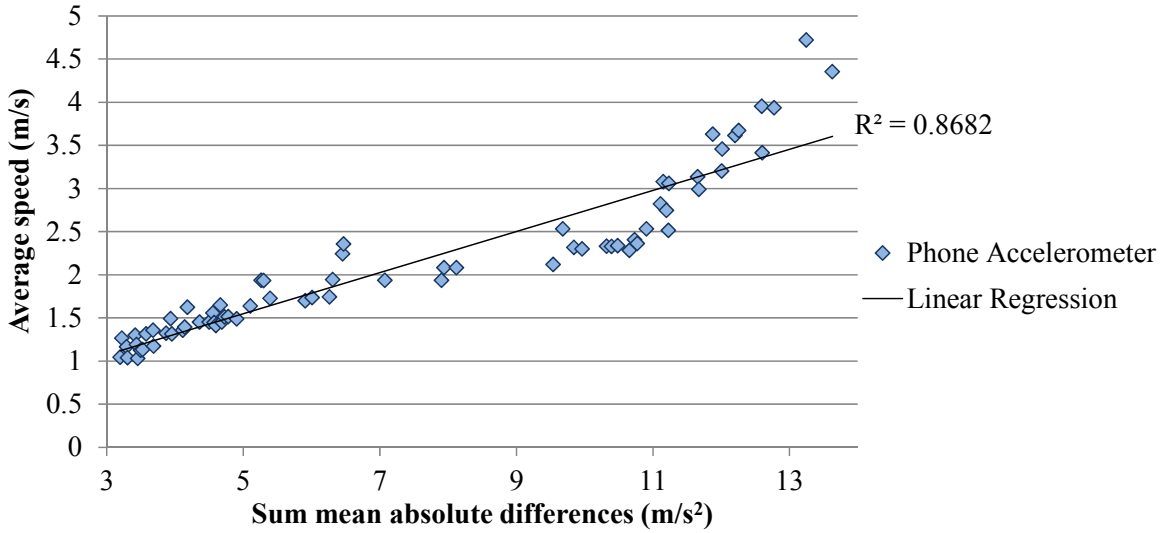


Figure 34. Scatter plot of the average speed versus sum of mean absolute differences on each axis of the phone accelerometer, including the corresponding linear regression.

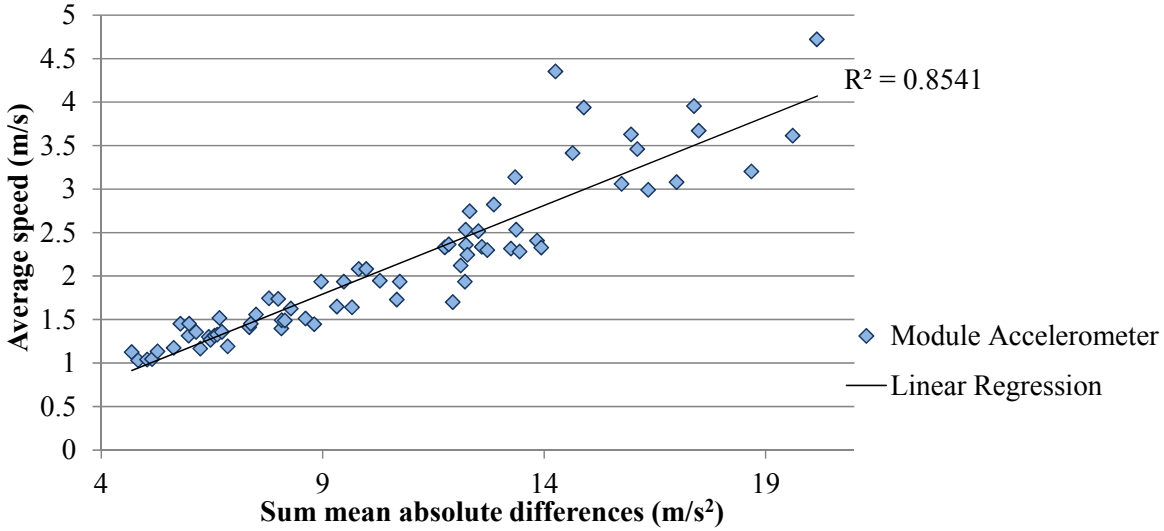


Figure 35. Scatter plot of the average speed versus sum of mean absolute differences on each axis of the module accelerometer, including the corresponding linear regression.

The remaining feature of three used for activity recognition, the sum of variances on each axis (4), was also analyzed for speed estimation purposes. The corresponding scatter plots for the phone and module accelerometers are shown in Figure 36 and Figure 37. When comparing these plots with the ones obtained for the sum of mean absolute differences on each axis (5), the main difference is the wider range of feature values. Concerning the associated coefficients of determination, the values of 0.8546 and 0.8735 are similar to the values of 0.8682 and 0.8541 obtained for (5).

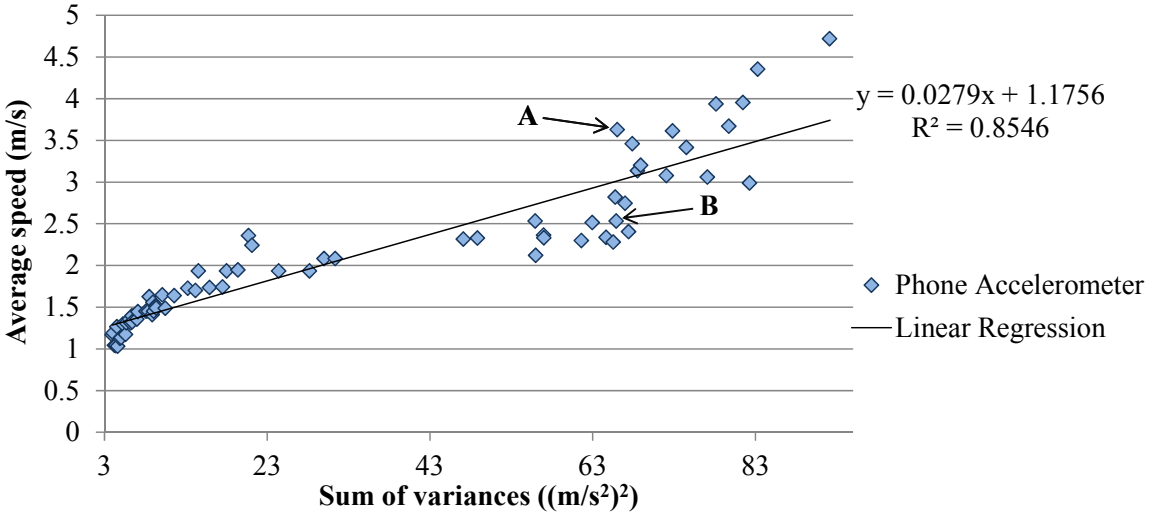


Figure 36. Scatter plot of the average speed versus the sum of variances on each axis of the phone accelerometer, including the corresponding linear regression. Points A and B refer to two different events. In event A a subject was running with higher speed than the speed corresponding to the subject in event B.

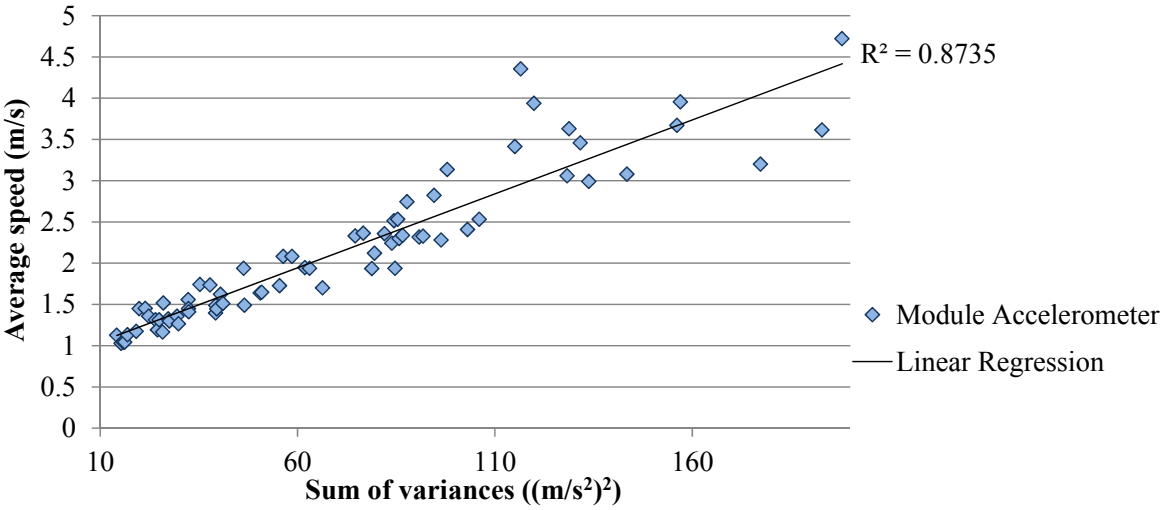


Figure 37. Scatter plot of the average speed versus the sum of variances on each axis of the module accelerometer, including the corresponding linear regression.

To better understand the big difference between the sum of mean squares and the sum of variances features, the points corresponding to events A and B (shown in Figure 32) are also shown in Figure 36. In this figure, it can be seen that the feature value for the higher speed subject (event A) is not much lower than the value lower speed subject (event B). In fact the corresponding speed of event A is now slightly higher than the speed corresponding to event B, as explained below. So the problem of a much lower feature value for the higher speed subject, pointed out above (Figure 32) is now partially attenuated.

The reason for the difference between the two features can be explained by observing Table 8. Considering the y-axis value, the higher speed event has a much lower value when the mean squares are used, but for the variances the values for the two events are not so markedly different. Furthermore, for the variance the x- and z-axis values are markedly greater for the higher speed subject. As a final result the sum of variances feature has higher values for the higher speed event (last column of Table 8) in contrast with the sum of mean squares feature where the higher speed subject has a significantly lower value. This result shows that the way a subject walks or runs will have an impact on the three accelerometer axis so all of them must be considered for good speed estimation.

Table 8. Feature values for two subjects corresponding to points A and B of Figure 32 and Figure 36

Feature	Event	Speed (m/s)	Feature per axis $((m/s^2)^2)$			Feature value $((m/s^2)^2)$
			x-axis	y-axis	z-axis	
Sum mean squares	A	3.62	14.77	91.66	7.26	113.70
	B	2.53	12.14	129.68	5.69	147.52
Sum variances	A	3.62	14.62	44.68	6.67	66.09
	B	2.33	8.57	54.87	2.47	65.92

Regarding the gyroscopes, for the module (Figure 39) the coefficient of determination (R^2 = 0.7811) is lower when compared to the last values presented above (for the sum of variances). For the phone gyroscope (Figure 38), although the coefficient of determination is similar to the phone accelerometer (0.8503 instead of 0.8546), it can be easily seen in Figure 38 that it is not adequate for estimating lower speeds.

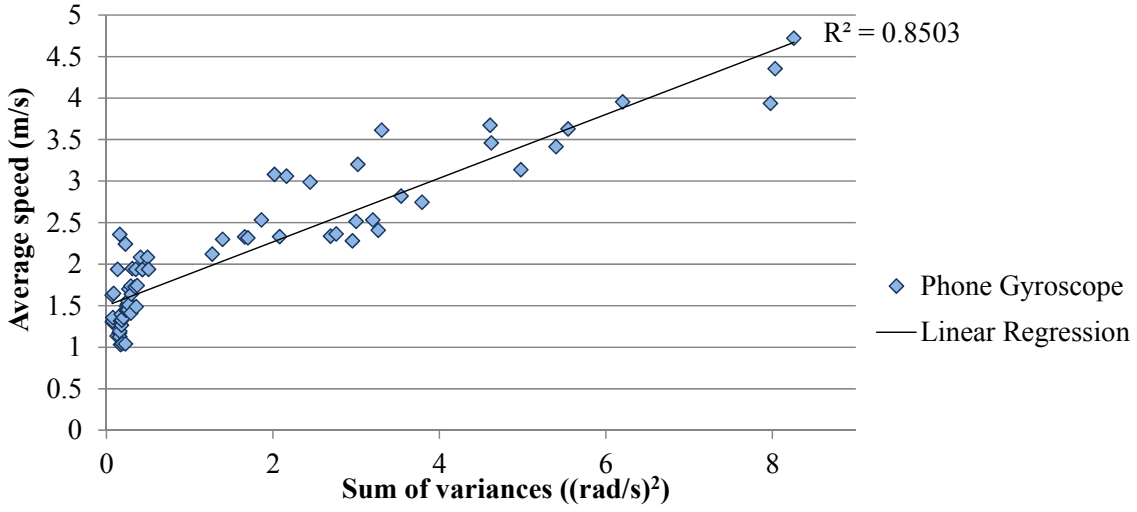


Figure 38. Scatter plot of the average speed versus the sum of variances on each axis of the phone gyroscope, including the corresponding linear regression.

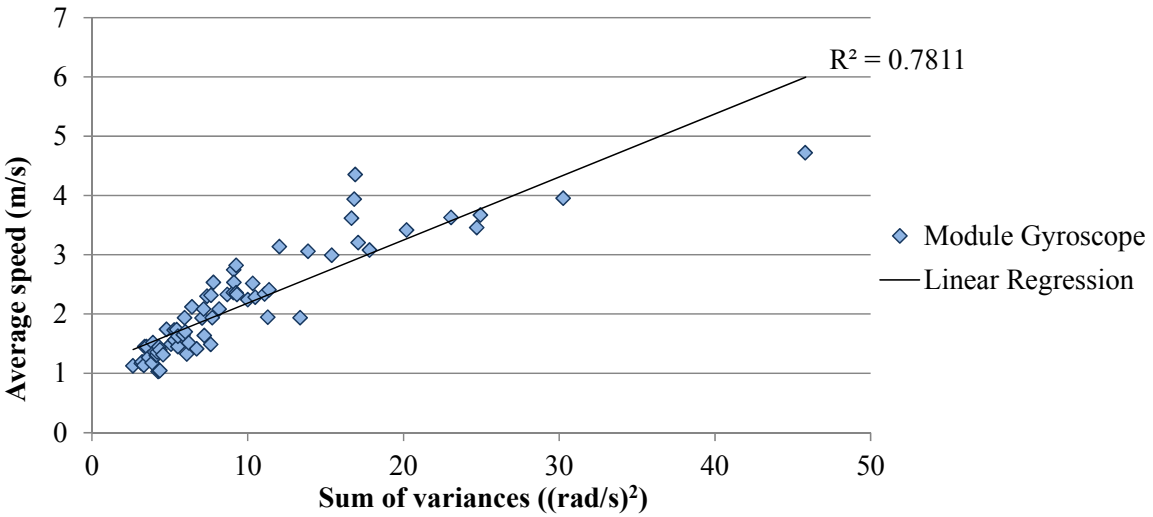


Figure 39. Scatter plot of the average speed versus the sum of variances on each axis of the module gyroscope, including the corresponding linear regression.

Considering all the features studied above for speed estimation, the sum of variances on each axis of the phone accelerometer was the chosen feature, since it has one of the best coefficients of determination values and is the same feature used in the activity recognition for distinguishing between static and dynamic activities, and between walking and running. So this choice is a great advantage regarding our aim of saving processing resources. Furthermore, it meets another aim of this dissertation, which is to explore the possibility of using only the smartphone.

So the equation of the linear regression included in Figure 36 (sum of variances for the phone accelerometer), is used to estimate the user speed. This equation can be written as (10), which gives the estimated average speed, in m/s, in function of the sum of variances for the phone accelerometer.

$$\mathbf{speed}\left(\boldsymbol{sum\_variances}\right) = \left(\mathbf{0.0279} \times \boldsymbol{sum\_variances} + \mathbf{1.1756}\right) \text{ m/s} \quad (10)$$

4.4 Experimental Results

Both algorithms described above (activity recognition and speed estimation) were evaluated using the data set aside for this purpose. The evaluations carried out and the corresponding results are presented in the next sections, followed by a discussion of the results and a comparison with some of the studies described in Chapter 2.

4.4.1 Activity Recognition

Evaluation

An evaluation of the activity recognition algorithm was performed using a single continuous file. This file resulted from the concatenation in a random order of all activities events (standing, sitting, lying, walking, running and fall) from the test set. Since each sliding window can contain values from two contiguous activities, whenever a given window includes two different activities, the one that occurred first was considered as the correct activity. The sensitivity, specificity and accuracy values associated with each activity were calculated, using (11), (12) and (13), respectively.

$$\text{sensitivity} = \frac{\#(\text{true positives})}{\#(\text{true positives} + \text{false negatives})} \tag{11}$$

$$\text{specificity} = \frac{\#(\text{true negatives})}{\#(\text{true negatives} + \text{false positives})} \tag{12}$$

$$\text{accuracy} = \frac{\#(\text{true positives} + \text{true negatives})}{\#(\text{true positives} + \text{false positives} + \text{false negatives} + \text{true negatives})} \tag{13}$$

To explain the meaning of the terms used in these equations, in the context of activity recognition, consider for example the standing activity. If the present window contains data from this activity, then:

- A true positive corresponds to the recognition of the standing activity.
- A false negative corresponds to the recognition of an activity other than standing.

Still considering the standing activity, if the present window contains data from an activity different from standing, then:

- A false positive corresponds to the identification of the standing activity.
- A true negative corresponds to the identification of activity different from the standing activity.

Table 9 shows the confusion matrix obtained for this evaluation, where the lines represent the actual activity and the columns the activities recognized by MonitorMe. The numeric values correspond to the number of windows evaluated (i.e. decisions made). The corresponding sensitivity, specificity and accuracy values are presented in Table 10.

Table 9. Confusion matrix when using the single continuous file with the activity events concatenated in random order *

	Standing	Sitting	Lying	Walking	Running	Fall
Standing	**536**	11	16	28	8	0
Sitting	17	**187**	4	2	2	0
Lying	18	2	**188**	8	2	0
Walking	10	0	4	**563**	4	0
Running	4	6	1	2	**129**	0
Fall	0	0	0	0	0	**44**

* Rows are the correct activity, columns the MonitorMe classification

Table 10. Sensitivity, specificity and accuracy for each activity, when using the single continuous file with the activity events concatenated in random order

	Sensitivity	Specificity	Accuracy
Standing	89.5%	96%	93.8%
Sitting	88.2%	98.8%	97.6%
Lying	86.2%	98.4%	97%
Walking	96.9%	96.8%	96.8%
Running	90.8%	99%	98.4%
Fall	100%	100%	100%
Overall	**93.4%**	**98.2%**	**97.3%**

For comparison of MonitorMe with other systems, a second evaluation of the algorithm was performed following the typical evaluation found in the state of the art, where the recognized activity for each window (containing data from only one activity) is compared with the actual activity [13, 22, 23]. So similarly to other studies, instead of performing the evaluation against a heterogeneous and random sequence of activities, each activity was considered separately. In other words, each processed window corresponded to only one activity. The confusion matrix for this evaluation is shown in Table 11, and the corresponding sensitivity, specificity and accuracy values are presented in Table 12.

Table 11. Confusion matrix when considering the activity events separately *

	Standing	Sitting	Lying	Walking	Running	Fall
Standing	**576**	0	0	0	0	0
Sitting	0	**202**	0	0	0	0
Lying	0	0	**211**	0	0	0
Walking	7	0	0	**560**	0	0
Running	0	0	0	0	**134**	0
Fall	0	0	0	0	0	**33**

* Rows are the correct activity, columns the MonitorMe classification

Table 12. Sensitivity, specificity and accuracy for each activity, when considering the activity events separately

	Sensitivity	Specificity	Accuracy
Standing	100%	99.4%	99.6%
Sitting	100%	100%	100%
Lying	100%	100%	100%
Walking	98.8%	100%	99.6%
Running	100%	100%	100%
Fall	100%	100%	100%
Overall	**99.8%**	**99.9%**	**99.9%**

Discussion

The first evaluation regarding activity recognition, presented above, uses an approach that takes into account the existence of transitions between the 6 recognizable activities. For the used sequence of randomly ordered activities, the windows containing data from more than one type of activity represented 10% of all windows. In this evaluation every fall is detected (100% accuracy) and the other activities have sensitivities in the range [86.2, 96.9] %, specificities in the range [96, 99] % and accuracies in the range [93.8, 98.4] %.

It is important to note that most of the false negatives are due to the method used for dealing with transitions, where the recognition of an activity is delayed for two windows (Section 4.2.2). On the

other hand, in real situations, transitions may have longer time durations. Also, there would probably be more confusion with the fall activity. Nevertheless, the first evaluation is closer to reality than the second evaluation, which was performed with the intention of comparison with other studies, as mentioned above.

In the second evaluation, the walking activity has a sensitivity of 98.8% and all other activities have 100% sensitivity. On the other hand, standing has 99.4% specificity and all other activities have 100% specificity. This leads to accuracies of 99.6% for standing and walking, and 100% for the rest. These better results were expected since the activities are considered separately, i.e. each window has only one associated activity.

The results presented above were achieved using a threshold-based activity recognition algorithm that uses only three time-domain features extracted from the accelerometer data. Frequency-domain features were not explored since they generally are more complex to compute and do not lead to much better accuracies when compared with some time-domain features [13]. Furthermore, the features used in MonitorMe have very low computational cost and storage requirements, as concluded by [13].

Although the gyroscope data were analyzed, the accelerometer data were enough to recognize the considered activities in this dissertation. More useful information could probably be obtained by performing integration and/or derivation of the gyroscope data, as in [27] and [28], but the associated extra mathematical operations would not be suitable for a mobile phone implementation. Also, this extra computational cost does not always contribute to the improvement of the algorithm accuracy, as we can see from [27] where the gyroscope data is integrated to calculate body angles together with accelerometer data. Indeed, the range of sensitivities obtained in the second evaluation of MonitorMe (]98, 100] %) are similar to the range of sensitivities achieved in [27] ([96, 100] %), when considering the same set of activities.

Using only accelerometer data, in [15] accuracies over 97% were achieved for some of the studied classifiers (using other techniques than threshold-based). However, the classifiers were trained for each user separately. The MonitorMe algorithm was developed based on all the data collected from the various users, as it is intended for general use, without prior user specific calibration.

In contrast with MonitorMe, in [22] the processing necessary for activity recognition takes place at a server, although the data is collected from sensors of a mobile phone. Moreover, the window size is of 10 seconds, which may not be appropriate for situations where it is important to have quickly the most updated activity information. Considering the classifier with best overall sensitivity from the 3 classifiers studied in [22], walking, jogging, sitting and standing have sensitivities of 91.7%, 98.3%, 95% and 91.9%, respectively. The sensitivities achieved (in the second evaluation) with MonitorMe are higher, with 99.6% for walking and 100% for running, sitting and standing.

The sensors configuration used in [23] is similar to MonitorMe, with an uni-axial (instead of tri-axial) accelerometer placed at the chest and another at the thigh. The set of activities is also similar, with the difference that walking is not distinguished from running (only locomotion is recognized) and falls are not considered. The authors developed a threshold-based algorithm based on the mean absolute deviation to distinguish static (standing, sitting and lying) from dynamic (locomotion) activities. In this dissertation the mean absolute difference on each axis was also studied but was not used for algorithm implementation. The sum of the variances on each axis was used instead,

since it results in a greater relative difference between the values for dynamic and static activities (as explained above in Section 4.2.2). Another difference is that posture identification in [23] is based on median values, while MonitorMe uses mean values. The evaluation of their algorithm led to sensitivities of 93% for sitting, 73% for standing and 91% for lying and locomotion. The MonitorMe algorithm achieved 100% sensitivity for all postures and 99.8% for locomotion (99.6% for walking and 100% for running) in the second evaluation. Apart from the different features used, the worse results of [23] can be due to the use of sensors with only one axis and the use of a lower sampling rate together with larger windows (10Hz and 10 seconds, respectively, instead of 25Hz and 1 second of MonitorMe).

4.4.2 Speed Estimation

Evaluation

The speed estimator (10) was also evaluated, using 30% of data (test set) from experiment 2, which includes 30 walking and running events at different speeds. Figure 40 presents the scatter plot of the average speed estimated by MonitorMe versus the measured average speed. The measured average speeds were obtained from the covered distance and elapsed time measured during the performance of each walking/running event. Each estimated average speed corresponds to the average of speeds estimated by MonitorMe for all windows of the considered walking/running event (14). It is possible to see from Figure 39 that the trend is of an increase of the estimated speed with the actual speed, with only some local deviation from this trend, as expected. The associated coefficient of determination is of R^2=0.8944.

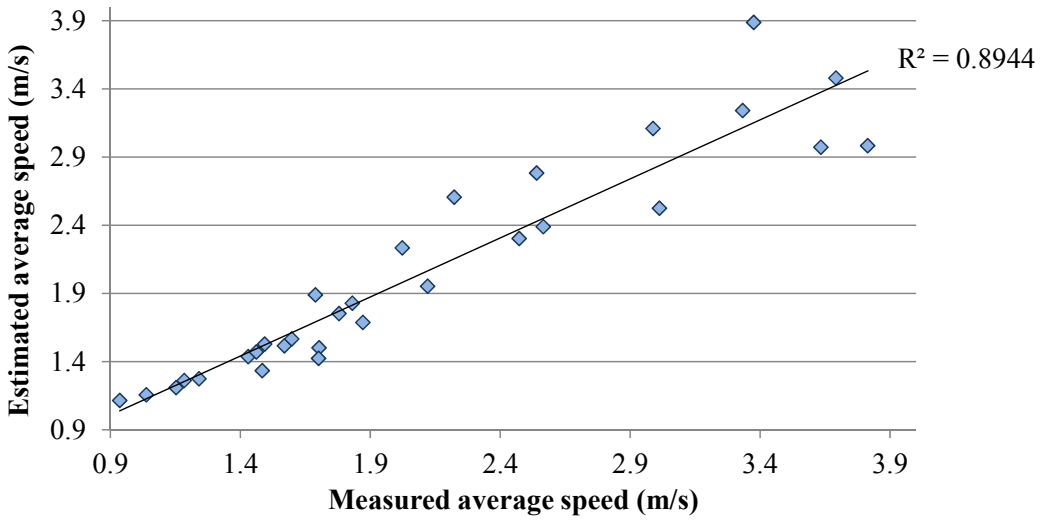


Figure 40. Estimated average speed versus the measured average speed for the test set of experiment 2 (walking and running).

Considering the measured and estimated average speeds, an average relative error of 8.6% was obtained using (15). The mean squared error (MSE), calculated using (16), resulted in a value of 0.0773 $(m/s)^2$.

$$\text{estimated average speed}_i = \frac{\sum_{j=1}^{W} \text{estimated speed}_j}{W}, \; W = \text{number of windows of event } i \tag{14}$$

$$\frac{\sum_{i=1}^{N} \left| \text{estimated average speed}_i - \text{measured average speed}_i \right| / \text{measured average speed}_i}{N}, \; N = 30 \text{ events} \tag{15}$$

$$\text{MSE} = \frac{\sum_{i=1}^{N} \left(\text{estimated average speed}_i - \text{measured average speed}_i \right)^2}{N}, \; N = 30 \text{ events} \tag{16}$$

Discussion

The evaluation of the speed estimator developed in this dissertation resulted in a coefficient of determination of 0.8944, an average relative error of 8.6% and a MSE of 0.0773 $(m/s)^2$, when considering the estimated average speeds and the corresponding measured average speeds. This confirms the suitability of (10) for estimating speed and therefore to control the video frame rate, when the user is walking or running.

In [9] the user speed was estimated by using exercise information that is generated and presented to the user. But the method used has the disadvantage of requiring the indication of the user age, sex and height for calculating the stride length based on a table of fixed values.

In [10] a different method is used for speed estimation, but it also relies on the user characteristics (height and weight). The speed estimator was developed using a training set obtained from data collected on a treadmill with controlled speed. It was then evaluated using the test set, resulting in a coefficient of determination of 0.975 and a MSE of 0.0225 $(m/s)^2$, when considering the actual and the estimated speeds, for a range of actual speeds between 1.31 a 4.76 m/s. A test with a single user running a distance of 10 km, in a track and field environment, resulted in a relative error of 9.57% between the actual average speed and the estimated average speed. The MonitorMe speed estimator was developed and evaluated from data collected in experiment 2, from a set of subjects with quite different ages, weights, heights and BMIs (values in Table 4), in a track and field environment. The evaluation resulted in a coefficient of determination of 0.8944, a MSE of 0.0773 $(m/s)^2$ and an average relative error of 8.6% between average estimated and average real speeds, for a range of actual speeds between 0.94 a 3.82 m/s. Besides the above considerations, to compare the two sets of results it is also important to note that the algorithm in [10] uses neural networks with a computational complexity difficult to evaluate from the presented data, but probably more complex than the MonitorMe algorithm. Furthermore, the latter is user independent (i.e. it can be used by any person and there is no need of personal information input). Taking this into account, the MonitorMe results can be considered good.

The acceleration values of both thighs were used together in [11] to estimate speed, using a third-order polynomial model obtained from the training set. In MonitorMe, the speed estimator uses a different feature extracted from the acceleration data of a single location (chest), instead of two locations (both thighs), and a first order polynomial model (more adequate for a mobile phone implementation). The sample rate and window size values are similar to the ones used in MonitorMe, as well as the number of subjects from which data was collected. However, in [11] a treadmill was used, while in MonitorMe the experiment took place in a track and field environment with many changes in wind intensity. Even though the model used in [11] is more complex and should provide a better fit, the MSE of 0.1358 $(m/s)^2$ is greater than the one obtained for the MonitorMe speed estimator evaluation (MSE = 0.0773 $(m/s)^2$).

In [8], data was collected from two bi-axial accelerometers (chest and right thigh). The RMS of the resultant acceleration was computed from these data, over 1-second windows, and was used for speed estimation. Just one subject was used for the study and no validation tests were performed with alternative users. In this dissertation data from accelerometers and gyroscopes placed at the same body locations were collected from 5 different users. From the study of these data it was concluded that the sum of variances on each axis, computed also over 1-second windows, provides better results compared with the mean squares feature. Further calculations have shown that the sum of variances is also better than the RMS feature (used in [8]).

5 MonitorMe Demonstration

In this section a simple demonstration trial of the MonitorMe system is presented and documented, to illustrate both the activity recognition and the video frame rate adaptability. MonitorMe is now fully deployed in an Android smartphone, a server and a remote device, in contrast with the early phases of the current work, where the development and evaluation stages were performed in a laptop. The setup presented in Figure **7** was used to show the MonitorMe online operation, using a laptop as both the server and the remote monitoring device for demonstration purposes. Also a wireless connection was used for communication between the smartphone and the server.

A 32 years old male subject, 1.8 m high and weighting 76 kg, performed at least once the standing, sitting, lying, walking, running and fall activities. This subject did not participate in any of the experiments carried out for the development of the MonitorMe algorithm. The activity recognition was performed online, with the current activity being sent to the remote device via the server. At the same time, the captured images (with variable frame rate) were also sent to the remote device.

For demonstration purposes, the following information was recorded for each window/decision: the timestamp of the moment when an activity was identified (*activity_timestamp*) and the name of the recognized activity. The value of the image frame rate, set according to the recognized activity, was also saved. For each captured image, the timestamp associated with the moment of capture (*image_timestamp*) was recorded.

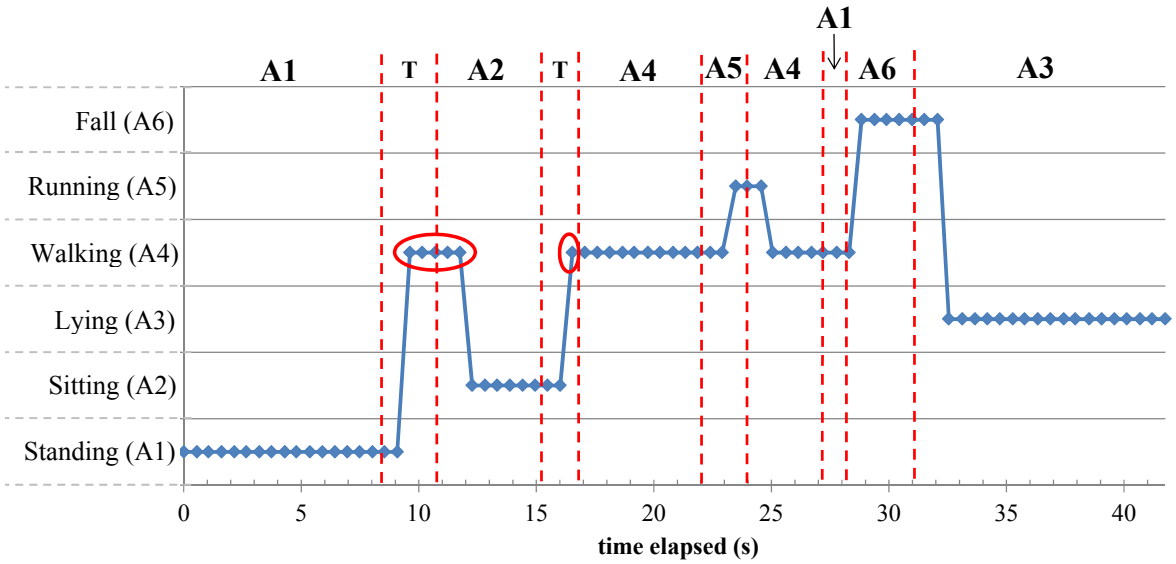


Figure 41. Activities recognized at each instant during the demonstration of the MonitorMe. The actual activities are shown above the graph, using the notation indicated in the graph y-axis. The red dashed lines indicate the duration of each actual activity. The read ellipses show incorrect recognition during transitions.

The graph presented in Figure 41 shows the activities recognized by MonitorMe as a function of the time elapsed since the beginning of the demonstration, obtained from the *activity_timestamp* values. The sequence of actual activities carried out is presented above the graph, together with

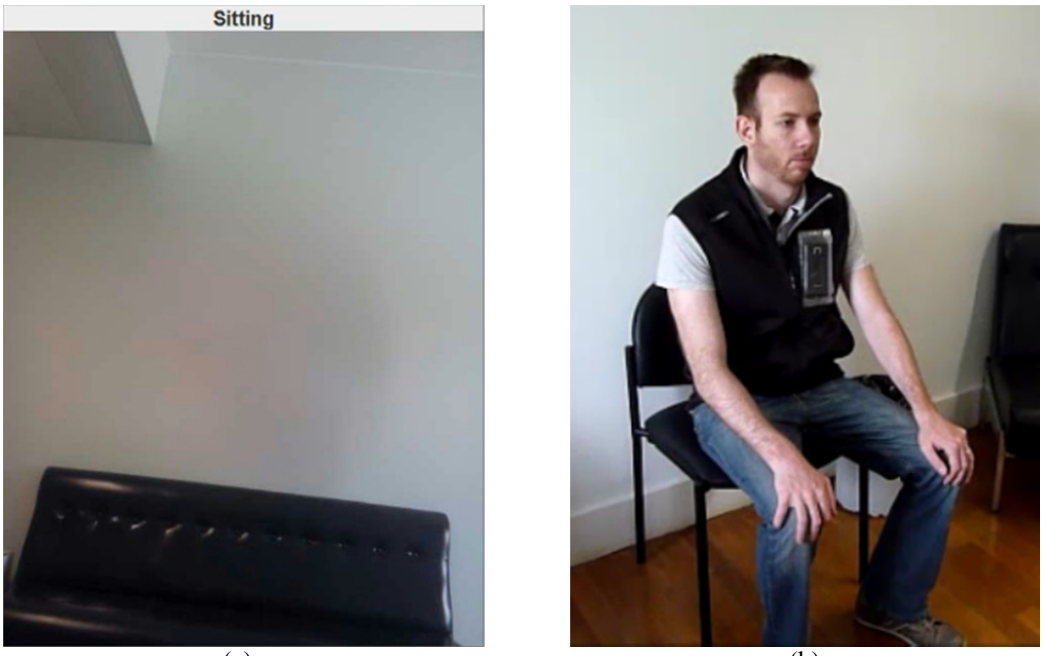
vertical red dashes lines that indicate the time intervals of each activity event. These activities were identified by using a video recorded during the demonstration. A1, A2, A3, A4, A5 and A6 refer to standing, sitting, lying, walking, running and fall, respectively. The marked transitions between activities are represented by the notation T.

For a proper interpretation of the demonstration, it is important to recall that the algorithm developed in this dissertation is not intended for detecting transitions between activities. As explained before, to deal with these events the algorithm maintains the previous decision until a new recognizable activity is detected for at least three consecutive windows. This approach works in many transition situations, but as explained previously, some situations still remain where transitions are recognized as an activity other than the activity performed before the transition (Section 4.2.2). These situations occurred in this demonstration, as indicated by the red ellipses in Figure 41, where marked transitions are recognized as walking. Depending on the application scenario, this may be of little consequence, since these false recognitions will occur only during time intervals of the order of a few seconds.

Apart from these transition situations, it can be seen in Figure 41 that all activities are correctly recognized with only a small delay, which was expected due to the method used to address transitions. Also due to this method, in the walking-standing-fall (A4-A1-A6) sequence, the standing activity (A1) is not recognized because the subject stood still for just about 1 second (duration of one window), between the walking activity and the fall. As the standing position was not recognized for more than two windows, the previous activity (walking) was kept, in accordance with the implemented algorithm.

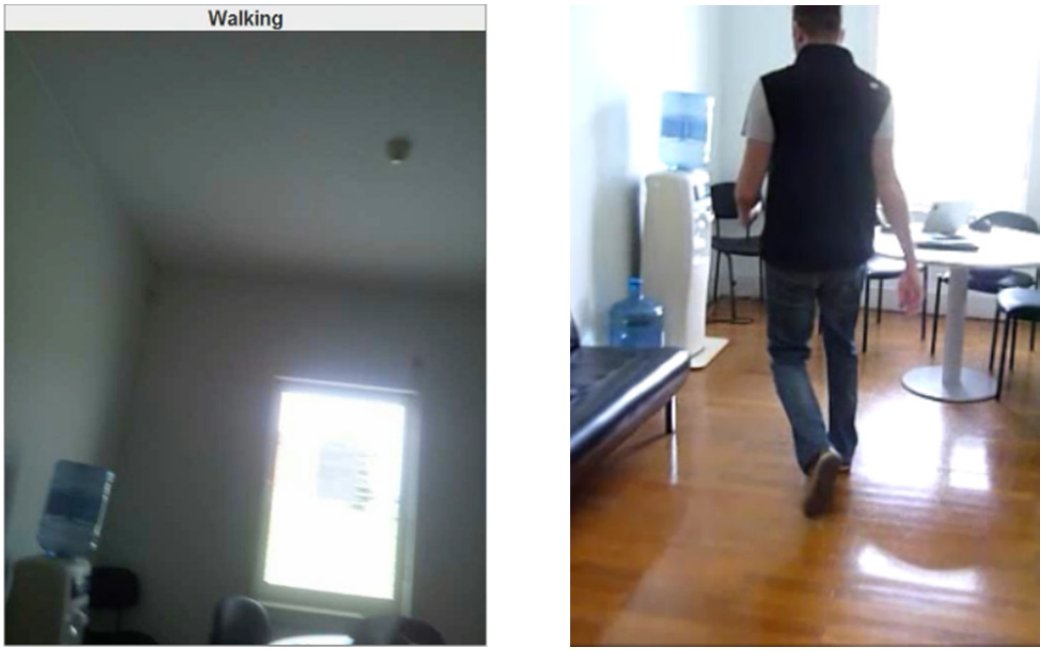
Regarding the fall activity, unlike other activities, it is recognized with almost no delay. This is due to the fact that the method to deal with transitions is not used when a fall is detected. Also, the fall detection is maintained during a longer time than would be expected, because the subject changed his position immediately after falling. So the next activity (lying) is only recognized when the user is lying still for a few seconds.

Figure 42 and Figure 43 show images captured during the demonstration, for the sitting and walking activities, respectively. Figures (a) present images that correspond to the information shown at the remote device, with the indication of the current activity being performed and current image of the user surrounding environment. Figures (b) show images of the user taken from the video recorded during the demonstration. These images were used to identify the actual activities carried out, to confirm the correctness of the activities recognized by MonitorMe.



(a) (b)

Figure 42. Capture from the remote device screen (a) and from the video used for identification of the actual activities (b), while the subject was sitting.



(a) (b)

Figure 43. Capture from the remote device screen (a) and from the video used for identification of the actual activities (b), while the subject was walking.

Figure 44 shows that MonitorMe adjusts the frame rate value according to the current recognized activity, figure (a), and that the video acquisition rate is effectively modulated using the frame rate value, as shown by the measured time intervals between consecutive frames, indicated in figure (b). It is important to note that although the ranges of elapsed time values are similar in figures (a) and (b), they do not correspond exactly to the same instants, i.e. they were obtained from different timestamps *(activity_timestamp* and *image_timestamp*, respectively). In figure (a) the recognized activities are shown above the graph, together with vertical red dashes lines that indicate the associated time intervals. A1, A2, A3, A4, A5 and A6 refer to standing, sitting, lying, walking, running and fall, respectively.

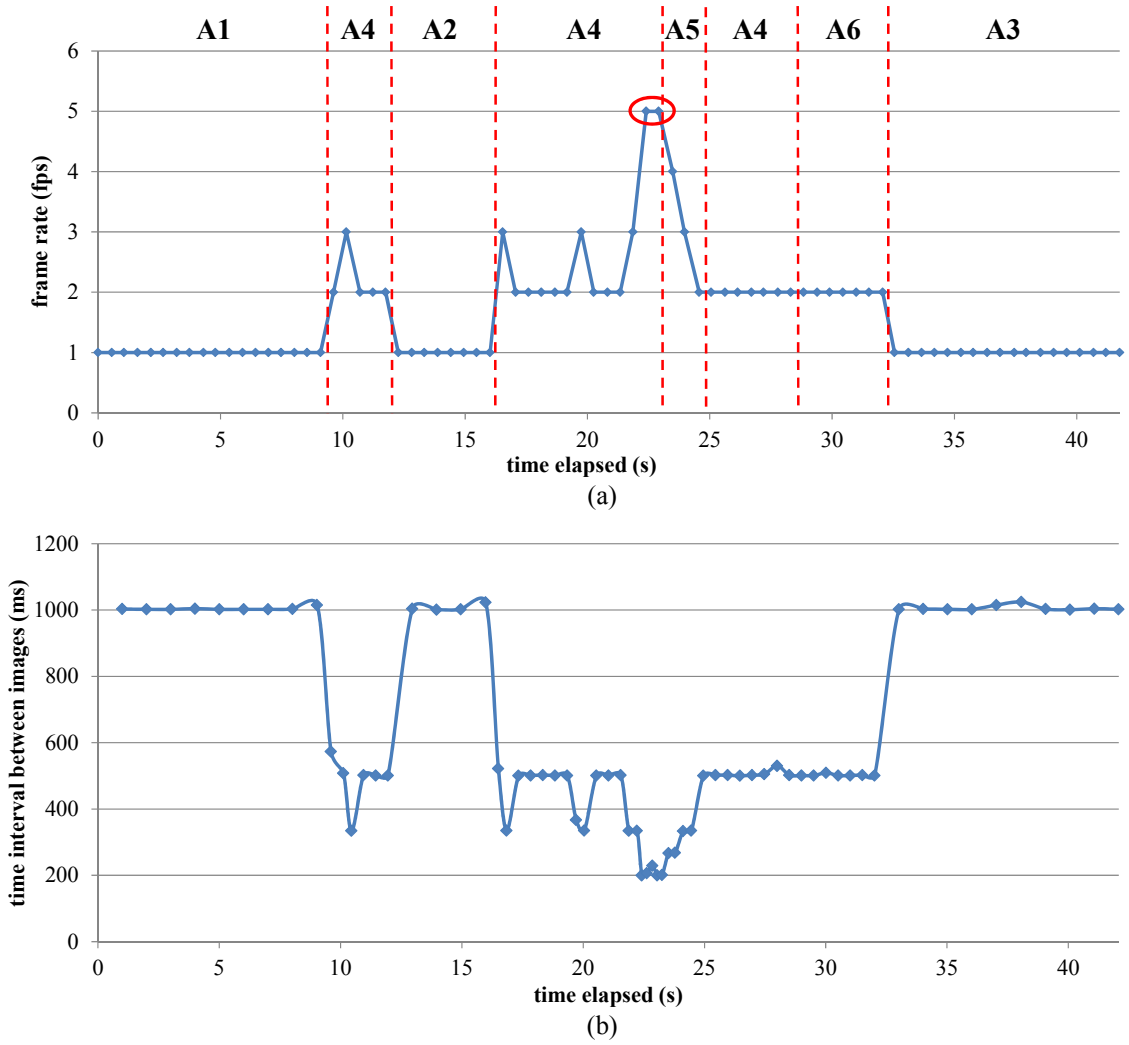


Figure 44. MonitorMe demonstration of the adaptable video frame rate: the frame rate value is computed according to activity/speed (a); this value is used to adjust the time interval between consecutive captured frames (b). The alphanumeric symbols on the top of figure (a) represent the activity being carried out in each time interval. The red ellipse corresponds to a case analyzed in detail in the text.

For a more detailed analysis, consider the sequence of activities (sitting, walking, running, walking) indicated in Figure 44 (a) by the sequence (A2, A4, A5, A4). It can be seen that when the user is sitting (A2) the frame rate is set to 1 fps, as expected. From the moment the walking activity

(A4) is detected, the frame rate is changed in the range between 2 and 3 fps. The 3 fps values correspond to faster speeds while walking.

Still relative to Figure 44 it is interesting to analyze the situation indicated by the red ellipse: at the end of the walking activity (A4) the frame rate value changes to 5 fps. To interpret this, it is necessary to take into account the delay associated with the recognition of a new activity, in order to deal with transitions (as mentioned above). So when the frame rate has the value of 5 fps, the user actually just started running, but the algorithm kept its previous decision of walking for two further processing windows. So, although the algorithm output is not yet running (A4 instead of A5), it makes sense that the frame rate is adapted to the effective higher speed that happens right before the actual running recognition.

During the time interval when the running activity (A5) is recognized, it is possible to see that the frame rate value decreases gradually from 5 fps to 2 fps. Considering the recognition delay explained above, this corresponds to the transition from running to walking, where the user speed decreases. While walking again (third A4 in the figure), the frame rate value stabilizes at 2 fps. After the considered sequence, it is possible to see that during the fall activity (A6) the frame rate value of 2 fps is maintained until the lying activity is recognized (the value returns then to 1 fps as expected).

Comparing plots (a) and (b) of Figure 44, it is possible to see that, as already mentioned above, the computed frame rate values were actually used when capturing images. For example, when the frame rate is set to 1 fps, the average time interval between consecutive images is of 1005 ms (corresponding to $1000/1005 = 0.995 \approx 1$ fps). Similarly, for 2, 3, 4 and 5 fps, the average time interval between captured images is of 506, 338, 268 and 207 ms, corresponding to 1.98, 2.96, 3.73 and 4.83 fps, respectively. The difference between the target and effective values can be explained by the fact that the frame rate of the phone camera (without any frame dropping) is not constant. However, the effective frame rate increases or decreases in accordance with the target values.

6 Conclusions and Future Work

6.1 Conclusions

In this dissertation we presented the MonitorMe system, which is able to monitor a subject by recognizing the activities being carried out, while simultaneously recording a video of the surrounding environment. Both the current recognized activity and the captured images are sent to a remote device. The activity recognition is based on sensor data collected from a smartphone and an in-house built MARG module. The video is captured as a sequence of images using the camera of the smartphone. The smartphone is placed at the user's chest (in a shirt pocket) and the MARG module at the right leg (in a pants pocket).

The MonitorMe smartphone application gathers and processes data from the accelerometers included in both devices, to perform online recognition of 6 different activities (standing, sitting, lying, walking, running and fall). This is achieved by using a low processing cost algorithm that requires only a small number of features (sum of minimum values, mean values and sum of variances).

The features used for activity recognition are extracted mainly from the phone accelerometer data. However, the results in this dissertation show that the smartphone alone would not enable the distinction between standing and sitting. This distinction is easily achieved with the MARG module data (placed at the leg). So this device is mandatory if the application scenario requires separate recognition of sitting and standing. Otherwise, the use of the smartphone at the chest is enough. Even though the implemented algorithm does not use gyroscope data, we observed that this sensor type could be also useful to distinguish static from dynamic activities, for example.

While recognizing activities, the phone additionally records a video of the surrounding environment in an efficient way, by adapting the frame rate in function of the estimated user speed. The non-GPS speed estimation is based on one of the features already computed for activity recognition. So the combination of simple speed estimation and frame rate adaptation allows further savings in data processing and required transmission bandwidth.

MonitorMe achieves similar or even better results than some alternative solutions available in the literature, considering comparable scenarios. Furthermore, when compared with other solutions, MonitorMe has the advantage of being an integrated system of activity recognition plus controllable image recording, which can be accessed remotely.

MonitorMe is an efficient, low computational cost system, which is an important characteristic in the development of smartphone applications with limited resources available. It is also a reliable solution that can be applied in many scenarios where mobility and autonomy are relevant. In health care, namely when involving elders, it is very important to provide autonomy. In the case of monitoring professionals in the field (e.g. first responders) it is also important to offer an autonomous solution with low signature (regarding space and weight) that can help in identifying critical or life threatening situations.

6.2 Future Work

The evaluation results of the MonitorMe algorithm presented in this dissertation are positive. However, a further validation is needed, given the limited number of subjects that participated in the experiments. This can be done namely by increasing the number of trials and subjects.

The main aim of this dissertation was to recognize the standing, sitting, lying, walking, running and fall activities. So transitions between these activities were addressed using the method explained in Section 4.2.2. However, it was verified that there are still some situations where transitions are confused incorrectly with a certain activity. A further analysis of sensor data is needed to deal with these situations, where other features associated with the accelerometer could be explored as well as the possibility of additionally use the gyroscope data. Also, a study similar to [20, 26] could be useful in the case of confusion with falls. The effect of the window size on the activity recognition results, especially in the case of transitions between activities, can also be studied.

Regarding the further reduction of the required processing power, some of the aspects that can be explored include the sampling rate value and the amount of window overlap. These can possibly be lowered without compromising the results obtained in this dissertation. Another way of minimizing the required processing and transmission bandwidth would be to start the capture and transmission of images only when certain activities are detected (e.g. fall, walking and running). Different methods for converting/compressing images can also be explored with the aim of reducing the processing time.

MonitorMe estimates speed by using information computed for activity recognition, instead of using the GPS that represents extra resources and can only be used outdoors. Nevertheless, the GPS can be useful in situations where it is important to know the user location. So if the user is outdoors, the GPS could be turned on when a certain event (e.g. a fall) occurs, to provide information about the user location.

In this dissertation a functional online monitoring system is presented, where the focus is on the smartphone application. For future work, this system could be improved by enabling the monitoring of several users. This could be applied in practical situations, such as team monitoring of fire-fighters. MonitorMe can also be integrated into the DroidJacket [50], an Android based monitoring solution developed in the context of the Vital Responder project, enhancing the monitoring of professionals such as fire-fighters.

The use of a mobile phone as a remote monitoring device could also be very useful, so an Android application for this device could be developed in the future by reusing the code of the Java application implemented in this dissertation. Another improvement, which could be especially important if a subject is monitoring a large number of users with a mobile phone, is the implementation of alarms for certain events. For example, if one of the users falls, an alarm can be sent to the mobile phone and a window opened with the information (activities and images) of the corresponding user.

References

[1] Merriam-Webster. (10 July 2012). *Definition of Monitor*. Available: http://www.learnersdictionary.com/search/monitor%5B2%5D

[2] United Nations, "The United Nations on World Population in 2300," *Population and Development Review,* vol. 30, pp. 181-187, 2004.

[3] World Health Organization. (10 July 2012). *Overweight and obesity*. Available: http://www.who.int/gho/ncd/risk_factors/overweight/en/

[4] International Association for the Study of Obesity. (10 July 2012). *About Obesity*. Available: http://www.iaso.org/policy/aboutobesity/

[5] V. F. Bento, V. T. Cruz, D. D. Ribeiro, and J. P. S. Cunha, "Towards a movement quantification system capable of automatic evaluation of upper limb motor function after neurological injury," in *Engineering in Medicine and Biology Society,EMBC, 2011 Annual International Conference of the IEEE*, 2011, pp. 5456-5460.

[6] Merriam-Webster. (10 July 2012). *Definition of Sensor*. Available: http://www.merriam-webster.com/dictionary/sensor

[7] Android Developers. (10 July 2012). *Sensors Overview*. Available: http://developer.android.com/guide/topics/sensors/sensors_overview.html

[8] J. Knight, H. Bristow, S. Anastopoulou, C. Baber, A. Schwirtz, and T. Arvanitis, "Uses of accelerometer data collected from a wearable system," *Personal and Ubiquitous Computing,* vol. 11, pp. 117-132, 2007.

[9] L. Myong-Woo, A. M. Khan, K. Ji-Hwan, C. Young-Sun, and K. Tae-Seong, "A single tri-axial accelerometer-based real-time personal life log system capable of activity classification and exercise information generation," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, 2010, pp. 1390-1393.

[10] Y. Song, S. Shin, S. Kim, D. Lee, and K. H. Lee, "Speed Estimation From a Tri-axial Accelerometer Using Neural Networks," in *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, 2007, pp. 3224-3227.

[11] W.-S. Yeoh, I. Pek, Y.-H. Yong, X. Chen, and A. B. Waluyo, "Ambulatory monitoring of human posture and walking speed using wearable accelerometer sensors," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, 2008, pp. 5184-5187.

[12] S. Yang and Q. Li, "Inertial Sensor-Based Methods in Walking Speed Estimation: A Systematic Review," *Sensors,* vol. 12, pp. 6102-6116, 2012.

[13] D. Figo, P. C. Diniz, D. R. Ferreira, and J. M. P. Cardoso, "Preprocessing techniques for context recognition from accelerometer data," *Personal and Ubiquitous Computing,* vol. 14, pp. 645-662, 2010.

[14] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, *Machine Learning, Neural and Statistical Classification*, 1994.

[15] A. Mannini and A. M. Sabatini, "Machine Learning Methods for Classifying Human Physical Activity from On-Body Accelerometers," *Sensors,* vol. 10, pp. 1154-1175, 2010.

[16] P. Casale, O. Pujol, and P. Radeva, "Human activity recognition from accelerometer data using a wearable device," presented at the Proceedings of the 5th Iberian conference on Pattern recognition and image analysis, Las Palmas de Gran Canaria, Spain, 2011.

[17] M. A. Ayu, T. Mantoro, A. F. A. Matin, and S. S. O. Basamh, "Recognizing user activity based on accelerometer data from a mobile phone," in *Computers & Informatics (ISCI), 2011 IEEE Symposium on*, 2011, pp. 617-621.

[18] L. Sian Lun and K. David, "Movement recognition using the accelerometer in smartphones," in *Future Network and Mobile Summit, 2010*, 2010, pp. 1-9.

[19] H. Ketabdar and M. Lyra, "System and methodology for using mobile phones in live remote monitoring of physical activities," in *Technology and Society (ISTAS), 2010 IEEE International Symposium on*, 2010, pp. 350-356.

[20] A. K. Bourke, J. V. O'brien, and G. M. Lyons, "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm," *Gait & posture,* vol. 26, pp. 194-199, 2007.

[21] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," *Pervasive Computing, Proceedings,* vol. 3001, pp. 1-17, 2004.

[22] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity Recognition using Cell Phone Accelerometers," presented at the 4th International Workshop on Knowledge Discovery from Sensor Data (SensorKDD-2010), Washington, DC, USA, 2010.

[23] K. Aminian, P. Robert, E. Buchser, B. Rutschmann, D. Hayoz, and M. Depairon, "Physical activity monitoring based on accelerometry: validation and comparison with video observation," *Medical and Biological Engineering and Computing,* vol. 37, pp. 304-308, 1999.

[24] P. H. Veltink, H. B. J. Bussmann, W. Vries de, W. L. J. Martens, and R. C. Lummel van, "Detection of static and dynamic activities using uniaxial accelerometers," *IEEE Transactions on Rehabilitation Engineering,* vol. 4, pp. 375-385, 1996.

[25] J. J. S. Martín, "Recognition of motion patterns based on mobile device sensor data," Master, Universität Stuttgart, Stuttgart, 2010.

[26] J. Jacob, T. Nguyen, D. Y. C. Lie, S. Zupancic, J. Bishara, A. Dentino, and R. E. Banister, "A fall detection study on the sensors placement location and a rule-based multi-thresholds algorithm using both accelerometer and gyroscopes," in *Fuzzy Systems (FUZZ), 2011 IEEE International Conference on*, 2011, pp. 666-671.

[27] Y. Zhang, S. Markovic, I. Sapir, R. C. Wagenaar, and T. D. C. Little, "Continuous functional activity monitoring based on wearable tri-axial accelerometer and gyroscope," in *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2011 5th International Conference on*, 2011, pp. 370-373.

[28] A. K. Bourke and G. M. Lyons, "A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor," *Medical Engineering & Physics,* vol. 30, pp. 84-90, Jan 2008.

[29] N. Cherniavsky, J. Chon, J. O. Wobbrock, R. E. Ladner, and E. A. Riskin, "Activity analysis enabling real-time video communication on mobile phones for deaf users," presented at the Proceedings of the 22nd annual ACM symposium on User interface software and technology, Victoria, BC, Canada, 2009.

[30] M. Mancuso and S. Battiato, "An introduction to the digital still camera technology," *ST Journal of System Research,* vol. 2, 2001.

[31] L. Freeman. (2012, 10 July 2012). *Cameras, sensors to help monitor patients' well-being at Physicians Regional hospitals*. Available: http://www.naplesnews.com/news/2012/mar/23/careview-camera-sensor-monitor-physicians-regional/

[32] CareView Commnunications. (10 July 2012). *SitterView*. Available: http://www.care-view.com/products.php?name=sitterview

[33] G. Dauphin and S. Khanfir, "Background suppression with low-resolution camera in the context of medication intake monitoring," in *Visual Information Processing (EUVIP), 2011 3rd European Workshop on*, 2011, pp. 128-133.

[34] Z. Zhongna, C. Xi, C. Yu-Chia, H. Zhihai, T. X. Han, and J. M. Keller, "Video-based activity monitoring for indoor environments," in *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, 2009, pp. 1449-1452.

[35] Contour. (10 July 2012). *Contour Connect*. Available: http://contour.com/software/mobile

[36] S. Mann, "'WearCam' (The wearable camera): personal imaging systems for long-term use in wearable tetherless computer-mediated reality and personal photo/videographic memory prosthesis," in *Wearable Computers, 1998. Digest of Papers. Second International Symposium on*, 1998, pp. 124-131.

[37] ZionEyez. (10 July 2012). *Eyez Specifications*. Available: http://www.zioneyez.com/#/specs/

[38] STMicroelectronics. (10 July 2012). *LIS3DH: MEMS digital output motion sensor ultra low-power high performance 3-axes "nano" accelerometer*. Available: http://www.st.com/internet/analog/product/250725.jsp

[39] STMicroelectronics. (10 July 2012). *L3G4200D (low-power three-axis angular rate sensor)*. Available: http://www.st.com/internet/analog/product/250373.jsp

[40] Samsung. (10 July 2012). *Galaxy S II - Especificações*. Available: http://www.samsung.com/pt/consumer/mobile-phone/smartphones/android/GT-I9100LKAOPT-spec

[41] GSM Arena. (10 July 2012). *Samsung I9100 Galaxy S II*. Available: http://www.gsmarena.com/samsung_i9100_galaxy_s_ii-3621.php

[42] V. F. Bento, V. T. Cruz, D. D. Ribeiro, M. M. Colunas, and J. P. S. Cunha, "The SWORD ambulatory rehabilitation system," in *33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC '11)*, Boston, USA, 2011.

[43] Microchip. (10 July 2012). *PIC24 16-bit Microcontroller*. Available: http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en026374

[44] Free2Move. (10 July 2012). *F2M03GLA (low power embedded Bluetooth module)*. Available: http://www.free2move.se/?page_id=911

[45] Kionix. (10 July 2012). *KXTF9 accelerometer*. Available: http://www.kionix.com/accelerometers/kxtf9

[46] InvenSense. (10 July 2012). *ITG-3200: Integrated triple-axis digital-output gyroscope*. Available: http://invensense.com/mems/gyro/itg3200.html

[47] Open Handset Alliance. (10 July 2012). *Android Overview*. Available: http://www.openhandsetalliance.com/android_overview.html

[48] Nielsen. (2012, 10 July 2012). *America's New Mobile Majority: a Look at Smartphone Owners in the U.S.* Available: http://blog.nielsen.com/nielsenwire/online_mobile/who-owns-smartphones-in-the-us/

[49] MathWorks. (10 July 2012). *Box Plots Documentation*. Available: http://www.mathworks.com/help/toolbox/stats/f4828.html

[50] M. F. M. Colunas, J. M. A. Fernandes, I. C. Oliveira, and J. P. S. Cunha, "Droid Jacket: Using an Android based smartphone for team monitoring," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*, 2011, pp. 2157-2161.